Snoopy Report Generator Snoopy2LaTeX

Internship Report *)

January, 2015 to July, 2015

Author:

Anjali Sharma

Information Technology,
Indian Institute of Information Technology, Allahabad, India

Supervisors:

Prof. Dr.-Ing Monika Heiner
Dipl. Inf. Christian Rohr

Department of Computer Science,
Brandenburg Technical University, Cottbus, Germany

*) Submitted and defended as Bachelor's thesis at the author's home university in November 2015.

Abstract

The aim of this thesis is to develop a new export functionality for Snoopy for the conversion of Snoopy-specific files into corresponding latex code that can be compiled by the user to generate human-readable reports.

The export functionality for Snoopy, i.e., Snoopy2LaTeX that is developed as part of this work, is inspired by SBML2LaTeX, an existing tool for conversion of SBML files into human-readable reports. The tool itself is available at:
`http://www.ra.cs.uni-tuebingen.de/software/SBML2LaTeX`

The tasks accomplished here can be broadly categorized as: (a) analysis of SBML2LaTeX and other existing Snoopy export functionalities, (b) iterative agile development of the report generator, and (c) comparison of the developed export tool with existing export functionalities.

The results render an added support for exporting all qualitative and quantitative net classes (including their colored variants, except Fault Trees) to latex code for generation of corresponding human-readable report that provides a detailed description of the net (by default) and renders user the functionality to customize each element of the report explicitly.

Keywords: Petri Nets, export, SBML, LaTeX, biochemical reaction networks

Contents

List of Figures

Task

Snoopy is a software tool to design and animate hierarchical graphs, among others Petri nets. It is extensively used for the verification of technical systems and validation of natural systems i.e. biochemical networks. **Latex** is on the other hand, a document markup language widely used for the preparation of scientific documents. A Latex-based compilation of nets developed using Snoopy, can thus be a revolutionary application. This functionality can render immense assistance to researchers and scientists for the preparation of scientific documents pertaining to their analysis of aforesaid natural and technical systems.

My task, thus, includes exploitation of this potential in the association of Snoopy with Latex and implementation of an Export functionality **Snoopy2LaTeX** that shall facilitate direct generation of human-readable reports for each net. This work is inspired by an existing tool **SBML2LATEX** and aims to extend the **Export to Latex** functionality currently present in Snoopy.

The essential steps that shall be taken are:

1. Analysis of SBML2LATEX, an existing file-format conversion tool,
2. Analysis of existing 'Export to Latex' functionality within Snoopy,
3. Compilation of a list of elements to export for each net class,
4. Design for the new export dialog with extensive customization options,
5. Template design for the expected output report (latex),
6. Code Implementation for the tool,
7. Review and testing of the tool,
8. Final documentation for the developed tool.

1 Introduction

Background
Modeling and analysis techniques have been widely used to study biochemical reaction networks. A large variety of modeling techniques are used to model these networks such as Boolean networks, Differential equations (ordinary or partial), Petri nets, etc. Among them, **Petri nets** are found to be a particularly suitable representation for these biochemical reaction networks. To gain some basic understanding about Petri net models, please refer [1].

At the *Chair of Data Structures and Software Dependability* at *Brandenburg University of Technology - Cottbus*, a modelling and simulation tool **Snoopy** has been developed. It is a tool to facilitate designing and animation of hierarchical graphs, among others types of Petri nets [1]. The tool is in use for the verification of technical systems, validation of natural systems, i.e. biochemical networks such as metabolic, signal transduction, gene regulatory networks etc. It is available for non-commercial use at: `http://www-dssz.informatik.tu-cottbus.de/snoopy.html`

Motivation
Snoopy is an efficient tool for modelling biochemical reaction networks and simulating them. It supports import and export functionality for various file formats to render executable results for a wide range of platforms, significant for any typical research work. The **Export to Latex** functionality in Snoopy currently renders a latex file corresponding to only the current view of the net created with Snoopy. It does not render any customization options or metadata about the net. The generated file, thus, is of little significance unless assisted by the description of the net, explicitly. Besides, the latex code cannot be directly used for analysis unless compiled by a TeX compiler.

My research and development work revolves around this challenge posed by 'Export to Latex'. The research work involves extensive analysis of this existing export functionality, while the development work includes implementation of a revised version of this functionality that shall enable Snoopy to generate not just the current net, but a complete human-readable report (pdf format) with all the net-specific and general metadata about the net, besides the option to generate corresponding latex file for the report. This shall render a much needed utility to generate a complete net documentation that can be directly used for scientific research work and analysis. The work is inspired by an existing tool SBML2LaTeX that converts SBML files to human-readable reports [3].

Outline
The research and development project pursued can be broadly outlined as a thorough analysis of the current version of SBML2LaTeX, Snoopy and its export functionalities followed by an iterative agile development of a new export functionality **Snoopy2LaTeX**. The task is concluded with a detailed documentation for this tool and its comparison with SBML2LaTeX.

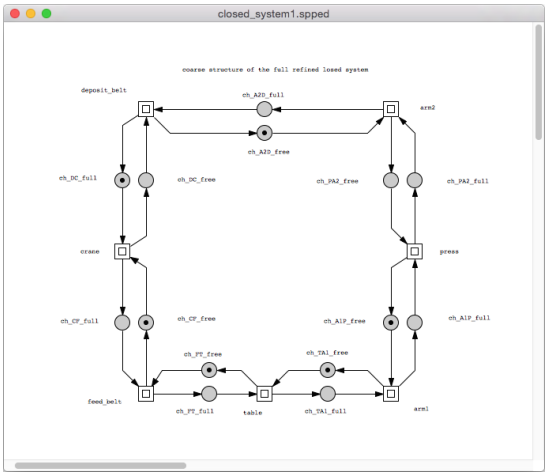
This thesis aims to render a detailed description of the development work pursued, besides providing an extensive record of all the technical, research and development issues encountered along the way; and the approach used to resolve them.

2 Problem Definition

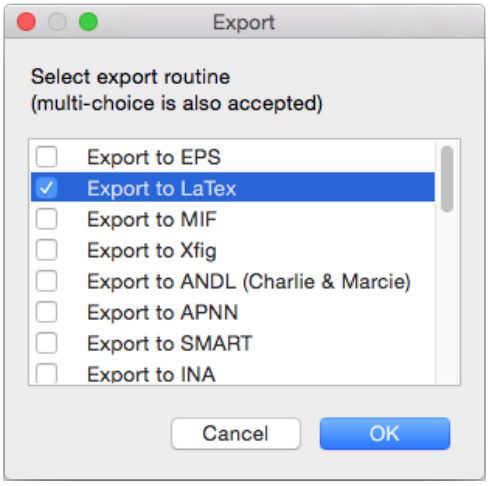
Among the many features and functionalities of Snoopy, it also provides efficient import and export for the Snoopy files from and into a wide range of file formats respectively. Currently, it facilitates exporting Snoopy file formats to several other file formats such as CSV, LaTeX, SBML, PRISM etc. besides the file formats specific to Snoopy, such as *pn* (Petri Net), *spn* (Stochastic Petri Net) etc. The various Exports offered by Snoopy are listed in Figure 1. It shows the allowed export formats for the corresponding graph classes listed to the right of each entry. Please refer to appendix B for details about available graph classes and file formats in Snoopy.

Figure 1: Exports in Snoopy. Screenshot from Snoopy website `http://www-dssz.informatik.tu-cottbus.de/snoopy.html`.

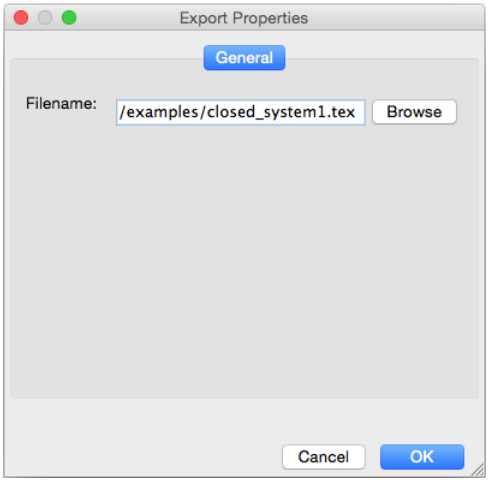
Export to Latex is one such export implemented in Snoopy that exports the Snoopy files to corresponding Latex code (extension for Latex files is .tex). The current 'Export to Latex' dialog is depicted in Figure 2.



(a) Petri net



(b) Export Dialog



(c) Export to Latex

Figure 2: Current Export Dialog. Screenshots from Snoopy 1.13. To export to latex, user needs to go to **File** → **Export** → **Export to Latex** and enter the required path for the output latex file.

As shown in Figure 2, the current export dialog includes just the option to browse the file name and proceed. Certain drawbacks in this current Export to Latex functionality are:

1. Only the currently active Petri net hierarchy level graph is exported into the corresponding Latex code. Thus, export becomes useless for hierarchical nets, since no the latex file provides no clue about the parent or child level associated to the net in the active window.

2. No information about the Petri net and the associated net class is provided. Without any metadata about the generated file, the latex code is obsolete.

3. No dynamic customization options. The user has no control over the layout of the exported file.

4. The generated file has to be explicitly compiled to generate the corresponding PDF file and put into use. Latex code of graph in itself cannot be directly interpreted unless compiled using a latex compiler

These issues faced by the current 'Export to Latex' functionality form the basis for my research and development work that aims to create a new export functionality that not only overcomes these drawbacks but also renders some added functionalities to the user. Some improvisations suggested are:

1. An improved export functionality that exports not only the active Petri net but the associated metadata to latex code
2. Options to export not only the net in the active window but all its sub-levels in a multilevel or hierarchical Petri net.
3. Customization options dynamic to the type of net class that allow user to customize the layout as well as content of the generated report or latex code
4. Option to compile the latex code from within the Export dialog using the latex compiler from user system.

These features, once implemented are likely to increase the utility of the current latex export manifold. The generated high-quality detailed reports can serve as an effective source for *model communication, proof-reading* and *error-detection.*

3 System Requirements

This section describes the basics hardware and software requirements that were considered for building and executing the Snoopy codebase. Certain specifications are liable for reconsideration or use of alternatives rendering the same functionality.

Hardware Requirements
No strict hardware constraints. They basically depend on your specific needs. For instance, if you plan to run big models (100,000 to 1000,000 variables), then higher requirements are needed than to run relatively small ones. The specifications for the system used for this development work are:

- **Machine:** Mac Pro (Early 2009)
- **Operating System:** OS X Yosemite (Version 10.10.1)
- **Processor:** 2 x 2,26 GHz Quad-Core Intel Xeon
- **Memory:** 8 GB 1066 MHz DDR3 ECC
- **Startup Disk:** Macintosh HD

Software Requirements
Except for the Snoopy-specific libraries used, all other specifications listed are liable for reconsideration based on availability of updates or alternatives.

- **IDE:** Eclipse (Indigo - Service Release 2)
- **Latex Editor:** TeXstudio 2.8.8
- **Latex Compiler:** pdflatex, latexmk
- **Libraries:**
  - wxWidgets 3.0.2 - C++ Cross-platform GUI Library
  - Snoopy Simulator (Snoopy-specific library)
  - SPSA_SteeringAPI (Snoopy-specific library)

4 Basic Methodology

Agile software development methodology was proposed and followed for the development of this new tool. In consideration to the cross-functional development team, need for quick results and iterative improvement of the tool, agile development methodology was apt for the work.

According to *Wikipedia*:

"Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement, and encourages rapid and flexible response to change."

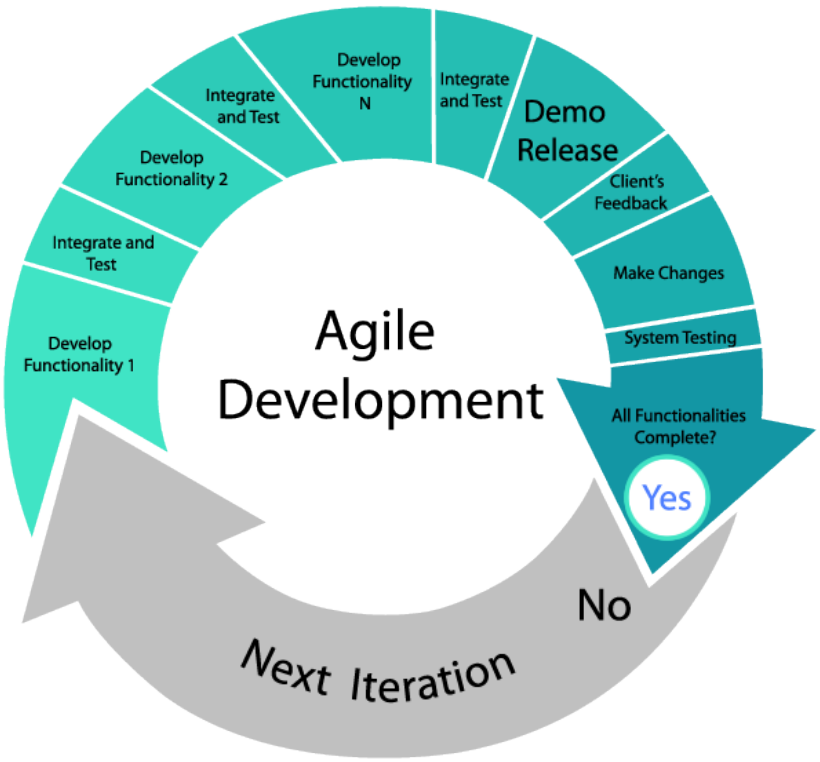


Figure 3: Agile Development Methodology. Source: `http://www.matrix-soft.org/agile-development-methodology/`

The tasks involved in the development process can be broadly defined as:

- **Analysis** of the Snoopy codebase and related existing tools
- **Design** for the new 'Export to Latex' dialogue
- **Development** of the new Export functionality
- **Testing** of the new Export functionality
- **Documentation** for the new report generator developed

The basic stepwise approach proposed to accomplish the above tasks can be given as:

1. Analysis
   (a) Analysis of petri nets
   (b) Analysis of Snoopy and its net classes
   (c) Analysis of existing 'Export to Latex ' functionality in snoopy
   (d) Analysis of the SBML2LATEX tool available for converting SBML files to human-readable reports
2. Design
   (a) Designing template for the Final report that shall be the output of the report generator
   (b) Designing template for the new Export Dialog with dynamic and static customisation options
3. Development
   (a) Code implementation for the approved Export Dialog User Interface
   (b) C++ implementation for the Latex code to be exported
4. Testing
   (a) Testing of the tool for different net classes
   (b) Testing of the tool for different OS platforms
5. Iterative improvisation of the template and implementation based on feedback
6. Documentation for the developed tool

The development of Snoopy2LaTeX Export tool follows the above workflow. The following sections include a detailed description of tasks carried out at each level of this workflow, in the same order.

5 Software Analysis

5.1 Petri Nets

A Petri net (also known as a place/transition net or P/T net) is one of the several mathematical modelling languages for the description of distributed systems. It is a directed bipartite graph, in which the nodes represent transitions (or events, signified by bars) of a reaction and places (or conditions, signified by circles). The directed arcs describe which places are pre- and/or postconditions for which transitions (signified by arrows) [7].

As discussed in [5], biochemical reaction systems have by their very nature three distinctive characteristics. (1) They are inherently bipartite, i.e. they consist of two types of game players, the *species* and their *interactions*. (2) They are inherently *concurrent*, i.e. several interactions can usually happen independently and in parallel. (3) They are inherently *stochastic*, i.e. the timing behaviour of the interactions is governed by stochastic laws. So it seems to be a natural choice to model and analyse them with a formal method, which shares exactly these distinctive characteristics: *stochastic Petri nets*.

The advantages of using Petri nets as a kind of umbrella formalism for systems and synthetic biology [5] are seen in the following:

- intuitive and executable modelling style,
- true concurrency (partial order) semantics, which may be lessened to interleaving semantics to simplify analyses,
- mathematically founded analysis techniques based on formal semantics,
- coverage of structural and behavioral properties as well as their relations,
- integration of qualitative and quantitative analysis techniques,
- reliable tool support.

Definition (Petri net, Syntax). *A* Petri net *is a quadruple $N = (P, T, f, m_0)$, where*

- *P and T are finite, non empty, and disjoint sets. P is the set of places (in the figures represented by circles). T is the set of* transitions *in the figures represented by rectangles).*
- *$f : ((P \times T) \bigcup (T \times P)) \rightarrow \mathbb{N}_0$ defines the set of directed arcs, weighted by nonnegative integer values.*
- *$m_0 : P \rightarrow \mathbb{N}_0$ gives the* initial marking.

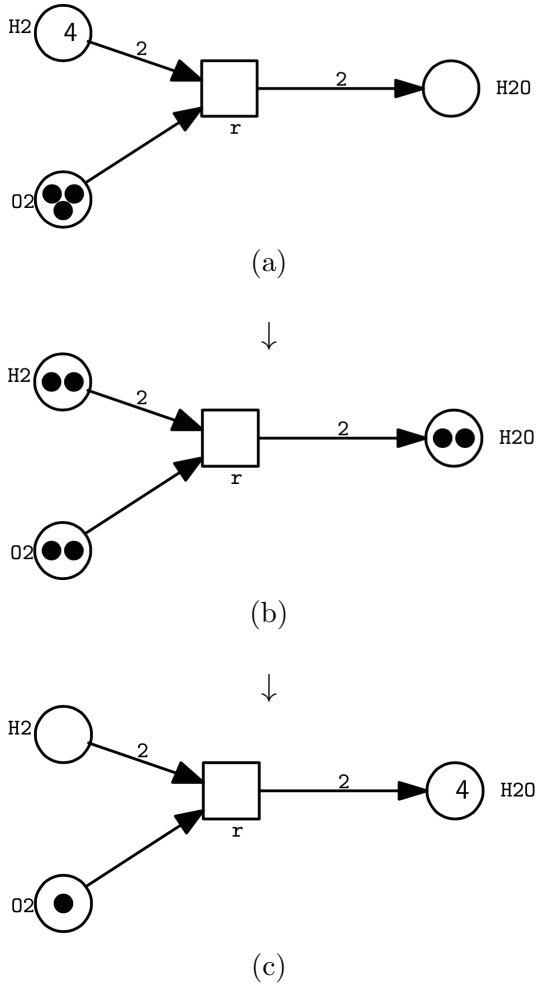


Figure 4: The Petri net for the well known chemical reaction r: $2H_2 + O_2 \rightarrow 2H_2O$ and three of its markings (states), connected each by a firing of the transition r. The transition is not enabled anymore in the marking reached after these two single firing steps.

Graphically, places in a Petri net may contain a discrete number of marks called *tokens*. Any distribution of tokens over the places will represent a configuration of the net called a *marking*. A transition of a Petri net may fire if it is *enabled*, i.e. there are sufficient tokens in all of its input places; when the transition fires, it consumes the required input tokens, and creates corresponding tokens in its output places. A firing is *atomic*, i.e., a single non-interruptible step [7].

Figure 4(a) represents Petri net for the chemical reaction: **r:** $2H_2 + O_2 \rightarrow 2H_2O$ with its initial marking. Places H_2 and O_2 contain 4 and 3 tokens, respectively. The tokens required for the transition are marked on the corresponding arcs. The default number of tokens required for a transition is considered 1 (as for the place O_2). When the transition fires, the required number of tokens are transferred to the output place H_2O as shown in Figure 4(b). Successive firing step leads to marking as shown in Figure 4(c). After this firing, the transition is not *enabled* anymore.

5.2 Snoopy

Systems biology is based on a trans-disciplinary joint effort to understand the complex mechanisms of life at the molecular systems level. For good reasons, experimentalists and theoreticians traditionally use different languages specific to their respective disciplines, thinking and expressing themselves in different ways. Experimentalists think in molecules and molecular mechanisms, which they illustrate with pictures, biochemical reaction pathways etc. Theoreticians communicate by using equations and mathematical symbols, which are difficult to read for the majority of experimentalists, who frequently do not have any significant mathematical background. A true understanding of each other would be greatly facilitated by establishing a communication platform (and language) which is equally easy to use for both experimentalists and theoreticians.

According to [4] that describes *Petri nets in Snoopy*, Snoopy is introduced as a software tool that supports the use of Petri nets as a

- platform or modelling language that can serve as a means of common communication for experimentalists and theoreticians
- framework that unifies the functionalities such as graphical display, computational modelling, simulation and bioinformatic annotation of biochemical networks, such as bacterial regulatory networks

Snoopy as per [2] is a software tool used to design and animate or simulate hierarchical graphs, among them the Petri net classes: standard Petri nets(PN), extended Petri nets (xPN), and extended Petri nets (xSPN). All the net classes supported by Snoopy are listed in appendix B.

Snoopy has three prominent features [2]:

1. **Extensible.** It has a generic design that facilitates the implementation of new graph types by the user.
2. **Adaptive.** It supports simultaneous use of several graph types, while the GUI adopts dynamically to the graph type corresponding to the active window.
3. **Platform-independent.** Snoopy runs on Mac OS X, Windows and Linux smoothly.

Figure 5 illustrates drawing of a Petri net in Snoopy. The software is freely available for non-profit academic research purposes at
`http://www-dssz.informatik.tu-cottbus.de/software/snoopy.html`.

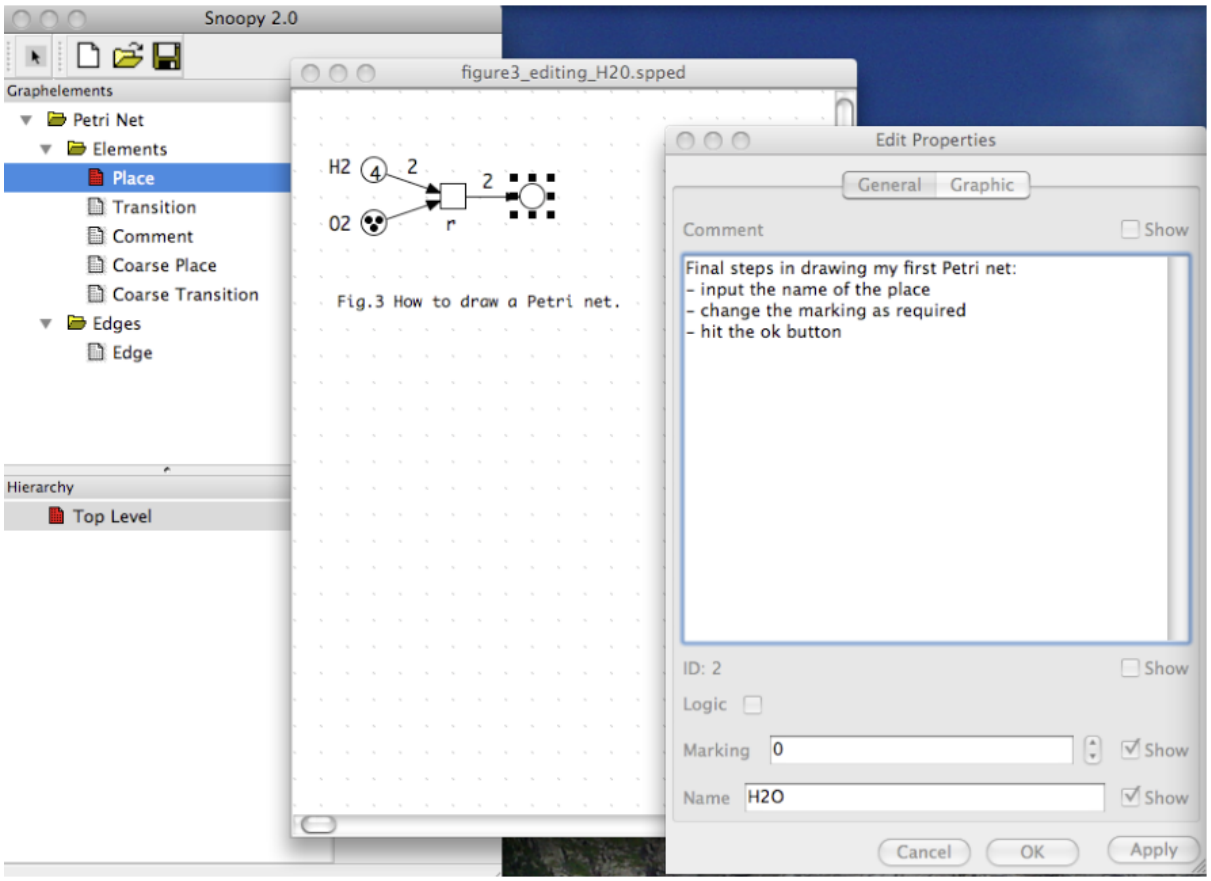


Figure 5: How to draw a Petri net with Snoopy. The menu panel on the left allows to select the type of graph elements to be created, the hierarchy panel to open macro nodes to show their sub-nets (not used here). The drawing window in the middle shows the Petri net under development. The window on the right belongs to the place that is selected in the drawing window and allows to edit the properties of this place. Taken from [4].

5.3 SBML2LaTeX

The Systems Biology Markup Language (SBML) has become the standard format for storing models of biochemical systems. Over 100 tools now support SBML, but certain important details like unit definitions, kinetic rate equations, user-defined functions, events etc. are not made explicit in the graphical presentations of these tools [3]. To detect potential errors or to gain overview of the model as a whole, it is necessary to examine content of the SBML file, but the unfriendliness of XML to human readers makes this an inconvenient and difficult task.

To address this problem, SBML2LaTeX was developed. It is a tool that accepts SBML files as an input and generates summaries of their content as reports in LaTeX source code format. For convenience of usage, an online web service directly produces human-readable files in various formats (illustrated in 5). Several settings provided allow for customization of the output, e.g. adding an extra title page instead of headlines, or choosing the paper size, orientation (landscape or portrait), font sizes and styles. All information is presented in clearly arranged tables, reaction equations and plain text, simplifying the task of understanding and communicating the model as well as detecting and correcting errors.

Figure 6: Example workflow using the SBML2LaTeX web service and SBML file BIOMD0000000003.xml (available at `http://www.ebi.ac.uk/biomodels`). After upload of an SBML file, several options allow customizing the output: MIRIAM annotations, an SBML consistency check or predefined unit declarations can be excluded, the desired file format can be selected; the paper size can be set to the US formats letter, legal or executive as well as to the European formats DIN A0-9 and the page orientation can be switched to landscape (especially important if the report contains fractions with very long denominators, where no automatic line break can be inserted). Several other options influence the layout of the report, e.g. names can be used in equations instead of identifiers, which can be displayed in typewriter or roman font. When the user clicks on the 'Convert' button, the report file is generated and accessible for download. Taken from [3].

The basic template design and functionality of our new Export to LaTeX is inspired by SBML2LaTeX. Besides, several additional features are introduced to this export. Section 9 presents a detailed analysis of SBML2LaTeX, while drawing its comparison with the developed Snoopy2LaTeX Export tool.

6 Template Design

6.1 Template Design for Export Dialog

Template design for Snoopy2LaTeX Export dialog is primarily inspired by the SBML2LaTeX export features as shown in Figure 27. This design for export dialog aims to:

1. Render an extensive list of **customization options that are dynamic** to the current net class being used.
2. Add features and custom options to the Export Dialog while keeping its look and feel **consistent with the existing infrastructure and design** of the overall export functionality for Snoopy.
3. Provide a set of **default settings** to the user that facilitate generation of a standard report for the Petri net, without much customization needed on part of the user.

Several drafts for this Export dialog were iteratively created and improved based on the feedback from the development team. The structure and features of the final template design (final draft used for implementation) are described here.

The Snoopy2LaTeX Export dialog is designed as a **tabbed document interface** (as illsutrated in Figure 7). This facilitates:

- the multiple export elements for customization to be separated into panels but at the same time be contained in a single window, rendering a **compact structure**.
- **easy navigation** across tabs, each specific to a different domain of export elements.

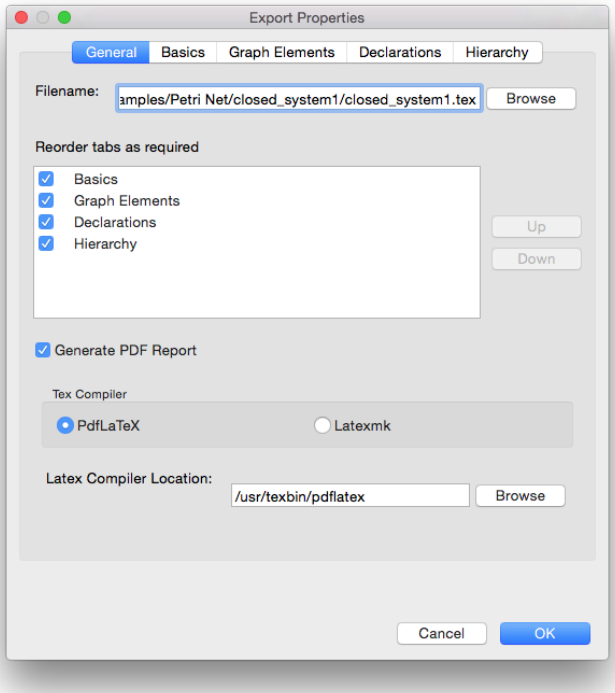
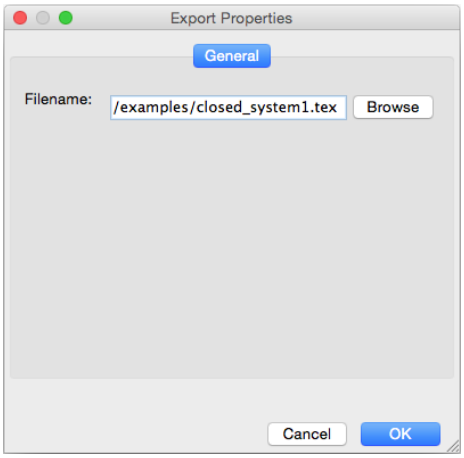


Figure 7: Snoopy2LaTeX Export Dialog: Tabbed Interface. The dialog is structured as 5 tabs (*General, Basics, Graph Elements, Declarations* and *Hierarchy*), each including custom options for a different domain of export elements.

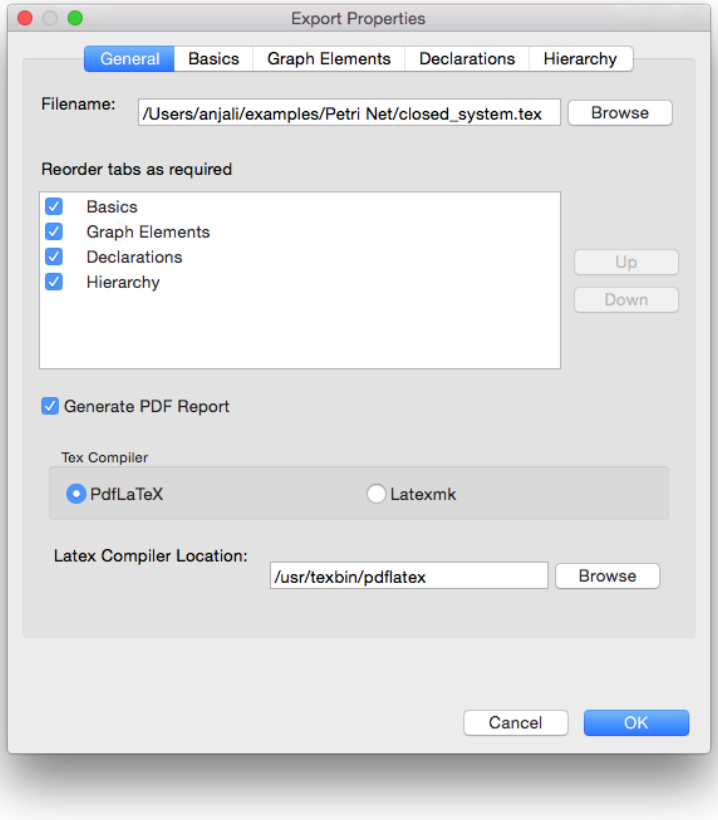
6.1.1 General Tab

This is the first tab in the dialog (as illustrated in Figure 8(b)). It corresponds to the general options for the export, which are *unrelated to the class or properties of the given net*. It is an extended version for *General* tab of the current version of Snoopy.



(a) General Tab: Current Version

↓



(b) General Tab: Redesigned Version

Figure 8: Template Design: **General Tab**. The current version of Snoopy has an Export to LaTeX dialog with a *General* tab with only a file picker (a). The new design offers several custom options besides the existing functionality (b).

Each element of this redesigned version is explained below:

- **'Filename' File Picker**

  This element is same as the one in the current Snoopy version. It is a file picker to specify the path of the output LaTeXfile. For the new Snoopy2LaTeX Export, several LaTeX are generated for a given net, and so this field is to specify the path for the main file among them.

- **Rearrange Panel**

  This panel lists the other four tabs, i.e., *Basics, Graph Elements, Declarations* and *Hierarchy* for reordering. The order of these tabs can be changed using the 'Up' and 'Down' buttons provided to the right of this panel. Besides, the tabs can be disabled for export by deselecting them from this list. The utility of this selection and ordering is explained in section 6.2.

- **'Generate PDF Report' Checkbox**

  Snoopy2LaTeX Export facilitates direct generation of PDF report by compilation of exported LaTeX code from within this dialog. This option can be enabled/disabled using the 'Generate PDF Report' checkbox. If deselected, only the LaTeX code for the given net is generated as output.

- **'TeX Compiler' Options**

  The radio button options in this field provide the option to switch between *pdlatex* and *latexmk* for generated LaTeX code compilation (if Export to PDF required).

- **'LaTeX Compiler Location' File Picker**

  This file picker is to provided to specify the path of the LaTeX compiler (as selected in the radio button group above) in the system. The path can be given via manual entry into the text field or by browsing using the **Browse** button.

For a detailed utility description of all these elements, kindly refer to section A.4.2.1 of *User Manual*.

6.1.2 Basics Tab

This is the second tab in the dialog (as illustrated in Figure 9). It lists the basic attributes of the net and layout & typography options for the output report, which are *common for any net class*.

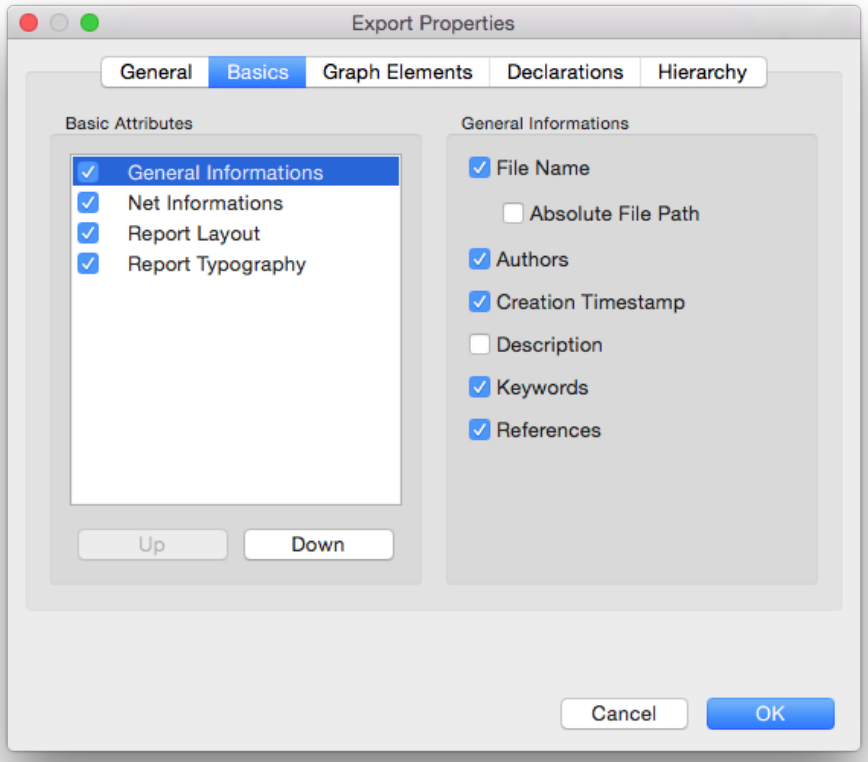


Figure 9: Template Design: **Basics Tab**

The Basics tab is organized into two panels.

- The **left panel** is a *rearrange panel* whose elements can be reordered using the 'Up' and 'Down' buttons provided below it. Also, the elements can be enabled/disabled for export by checking/unchecking them from the list, respectively. The utility of this selection and ordering is explained in section 6.2. This panel lists the basic elements for customization which include:
  General Informations, Net Informations, Report Layout and *Report Typography*.
- The **right panel** is *dynamic* to the current selection in the left panel. It lists the attributes for element selected in the left panel, for customization.

The design for the right panel corresponding to each basic element is presented in the following paragraphs.

1. **General Informations**

   This element corresponds to the 'General Informations' menu option of Snoopy (as illustrated in Figure 10). The attributes, as fetched, are added to the right panel as a checklist. This provides an efficient way for selection of attributes to be exported. These attributes include:

   (a) **Filename**

   It corresponds to the name of Snoopy net file (e.g. closed_system.spped).

   (b) **Absolute File Path**

   This gives the option to export the complete path of the net file in user system (e.g. /Users/anjali/Petri Net/closed_system.spped).

   (c) **Authors**

   It corresponds to 'Authors' field of General Informations dialog. It gives the option to export the author names (as specified in Snoopy's General Informations dialog) to the report.

   (d) **Creation Timestamp**

   It corresponds to the timestamp of net creation. If selected, the timestamp (*date in YYYY-MM-DD format and time in HH:MM:SS format*) is exported to the report.

   (e) **Description**

   It corresponds to the 'Description' field of General Informations dialog. It shall export the description for the net, as specified in this field.

   (f) **Keywords**

   It corresponds to the 'Keywords' field of General Informations dialog. It shall export the keywords for the net, as specified in this field.

   (g) **References**

   It corresponds to the 'References' field of General Informations dialog. It shall export the references for the net, as specified in this field.

2. **Net Informations**

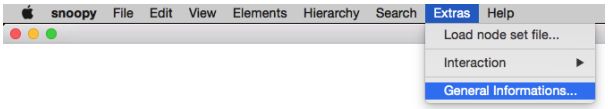
   This element corresponds to the 'Net Informations' menu option of Snoopy (as illustrated in Figure 11). Similar to General Informations, the attributes are added to the right panel as a checklist. These include:

   (a) **Net Class**

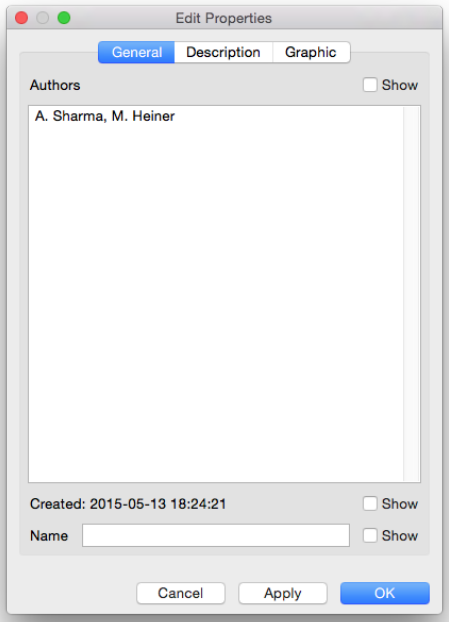
   It corresponds to all net classes (all *Node classes* and *Edge classes*) for the given net.

   (b) **Element Count**

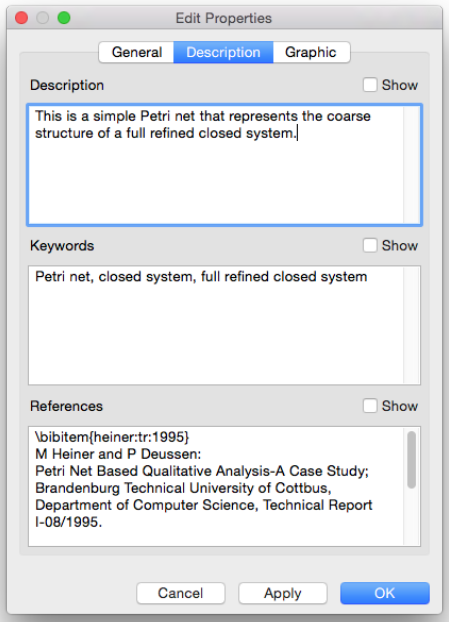
   It corresponds to the count of elements for each net class for the given net.



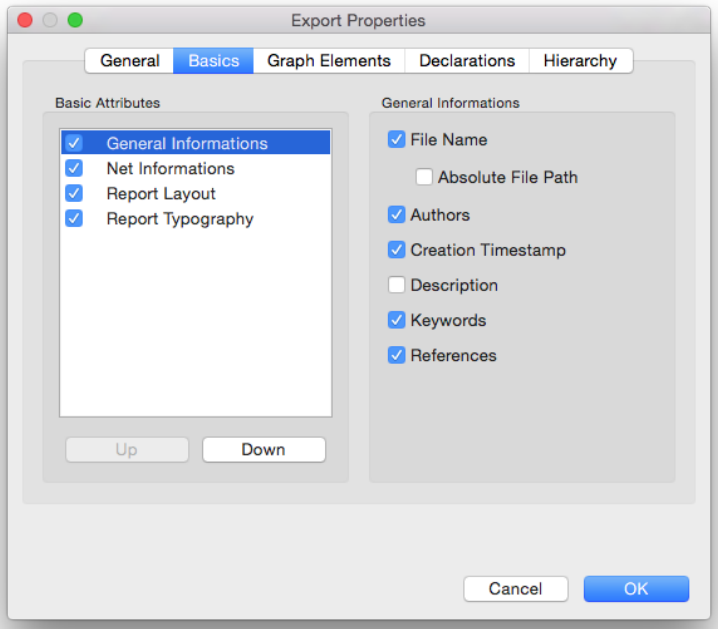
(a) Snoopy Menu: General Informations



(b) General Tab

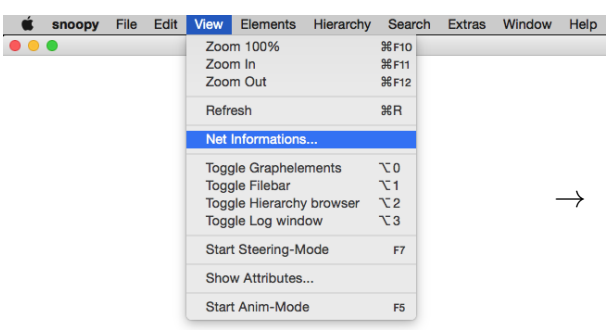


(c) Description Tab

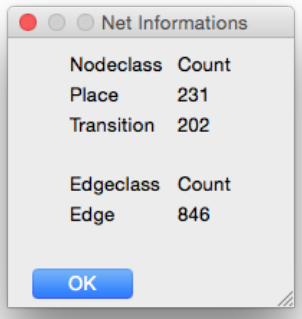


(d) Fetched General Informations

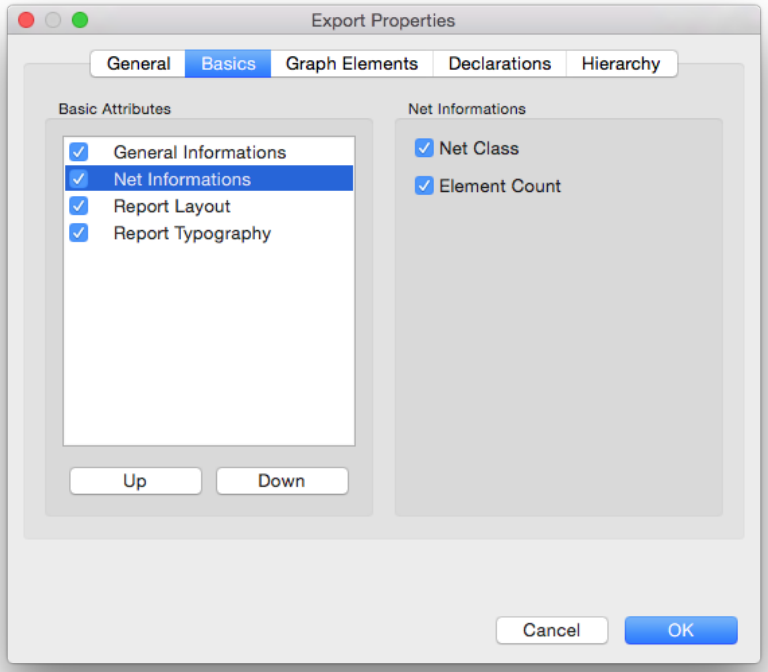
Figure 10: Template Design: **General Informations**. The General Informations option in the Snoopy menu (a) leads to dialog with tabs as depicted in (b) and (c). Our design (d) fetches the export attributes from the contents of this dialog.



(a) Snoopy Menu: Net Informations



(b) Net Informations Dialog



(c) Fetched Net Informations

Figure 11: Template Design: **Net Informations**. The Net Informations option in the Snoopy menu (a) leads to a dialog (b). Our design (c) fetches the export attributes from the contents of this dialog.

3. **Report Layout**

   This element corresponds to a set of custom attributes that define the layout of the generated report (as illustrated in Figure 12). While the orientation is provided as a radio button option, all other attributes are added to the right panel as a checklist.

These include:

(a) **Orientation**

It defines the page orientation for the generated report. It can be switched between *Portrait* and *Landscape* as desired by the user.

(b) **Insert Page Numbers**

It adds page numbering to the report as a footer.
(*Format: Page <Current Page> of <Total Page Count>*)

(c) **Insert Date**

It inserts current date (system date when the net is exported) to the report as a footer.
(*Format: DD/MM/YYYY*)

(d) **Insert Time**

It inserts current time (system time when the net is exported) to the report as a footer.
(*Format: HH:MM*)

(e) **Header**

It includes a text field to specify a header (filename, by default) for the report.

(f) **Footer**

It includes a text field to specify a footer (author(s), by default) for the report.

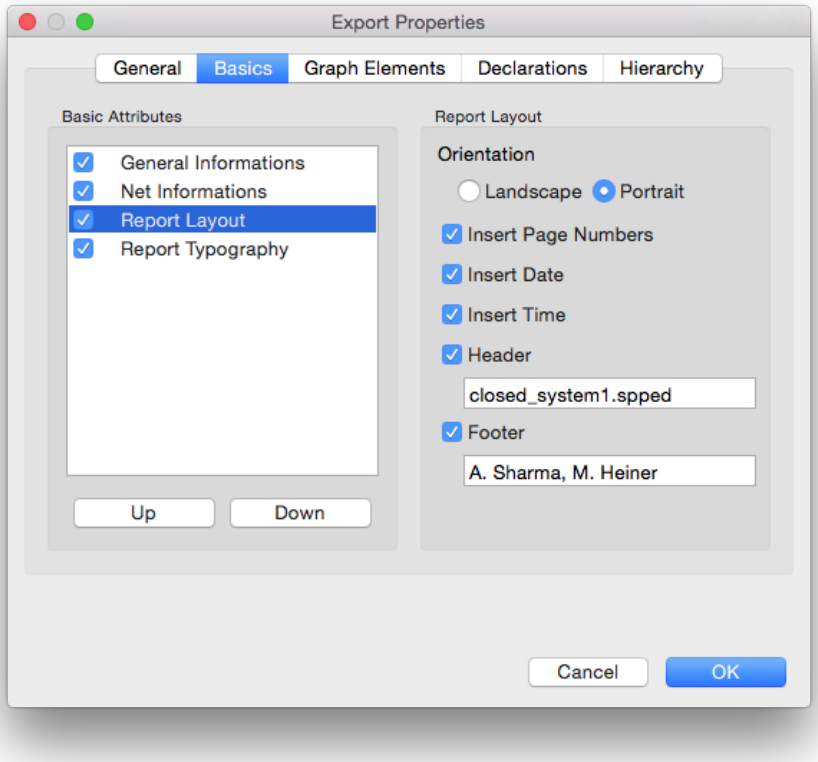


Figure 12: Template Design: **Report Layout**

4. **Report Typography**

   This element corresponds to a set of custom attributes that define the typography of the generated report (as illustrated in Figure 13). Each attribute has a range of options as an expandable list to chose from. These attributes include:

   (a) **Font Family**

   It allows the user to chose the font family (*roman, sans serif* or *monospace*) to be used for the LaTeX report, from the list. By default, *roman* is used.

   (b) **Paper Size**

   It allows the user to specify the paper size (*a4*, by default) for the LaTeX report. The user can chose from the range of pre-defined page sizes (specific to LaTeX) listed.

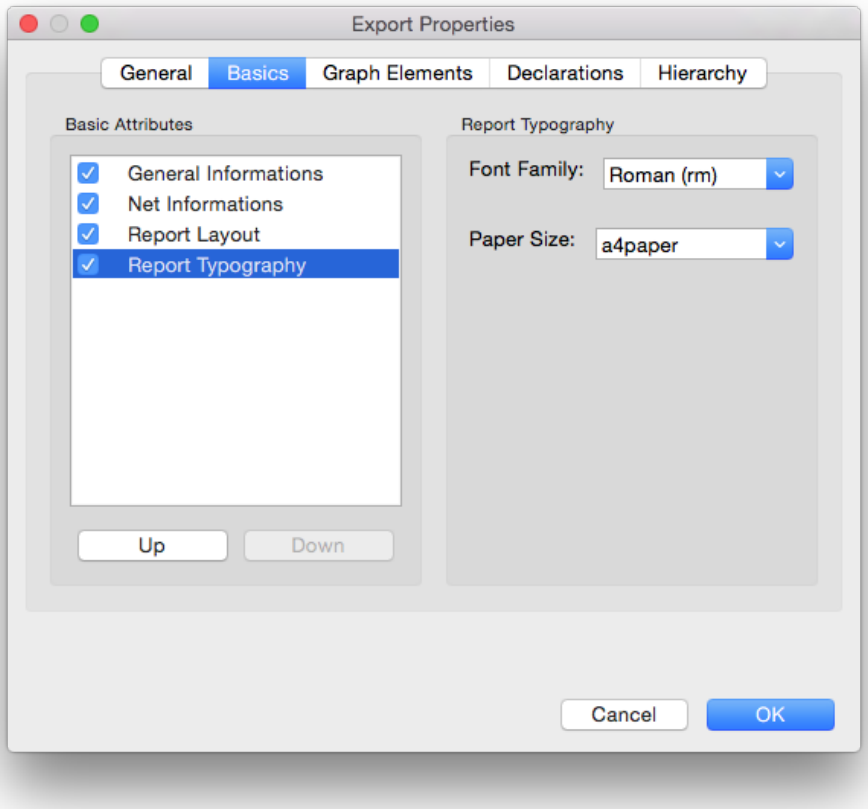


Figure 13: Template Design: **Report Typography**

For a detailed utility description of all these elements, kindly refer to section A.4.2.2 of *User Manual*.

6.1.3 Graph Elements Tab

This is the third tab in the dialog (as illustrated in Figure 14). It lists the graph elements for the given net and custom attributes for each element. Every net class has a unique set of graph elements. Hence, the contents of this tab are completely *dynamic to the net class for the given net.*

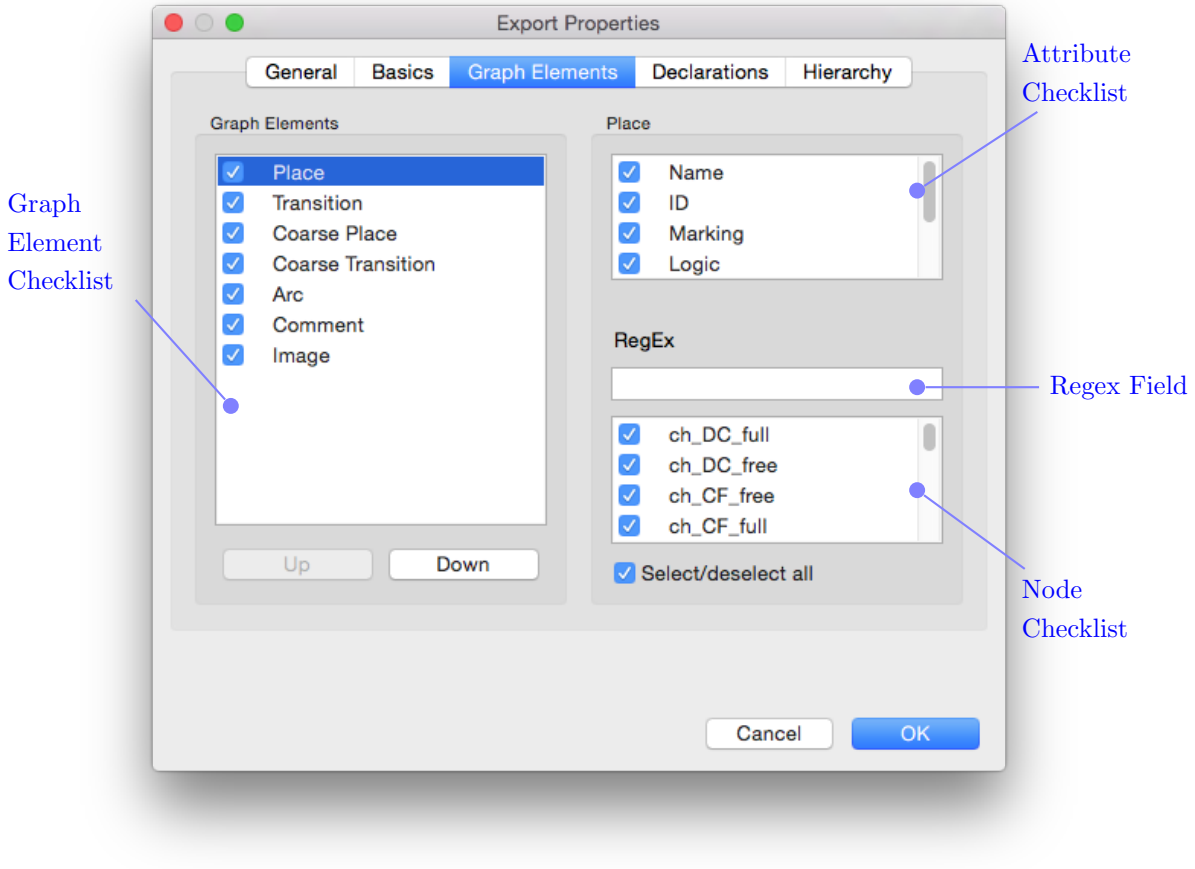


Figure 14: Template Design: **Graph Elements Tab**

The Graph Elements tab, like the Basics tab is organized into two panels.

- The **left panel** is a *rearrange panel*, as in Basics tab. Thus, elements can be enabled/disabled or even reordered. The utility of this selection and ordering is explained in section 6.2. This panel lists the graph elements for the given net.
- The **right panel** is *dynamic* to the current selection of graph element in the left panel. It lists the attributes and nodes corresponding to the selected element, for customization.

The generic structure of this tab is categorized into four main parts:

1. **Graph Element Checklist**

   Each net class has a unique set of graph elements. The left panel contains a checklist of graph elements for the given net, which can be individually selected/deselected or reordered. For example, in Figure 14, the graph elements for a net belonging to Petri net class are listed.

2. **Attribute Checklist**

   Every graph element has a unique set of attributes. This top right panel contains the attributes for the current selection of graph element, as a checklist. Any attribute can thus be selected/deselected for export by checking/unchecking the respective item in the list. For example, in Figure 14, the attributes for 'Place' element for are listed.

3. **Node Checklist**

   Every net has a unique set of nodes corresponding to each graph element. This bottom right panel contains the nodes belonging to the current selection of graph element, as a checklist. Any node can thus be selected/deselected for export by checking/unchecking the respective item in the list. Besides, a **Select/deselect all** checkbox is provided to select/deselect the complete node list for export. For example, in Figure 14, the nodes of 'Place' element type for the given net are listed.

4. **Regex Field**

   The **RegEx** field is provided to specify a regex (or regular expression). As soon as a valid regex is entered in the field, the node names in the 'Node Checklist' corresponding to this expression get selected, deselecting others. *To learn more about regex, please refer section A.6.*

For **Arcs**, the layout of this tab varies a bit (shown in Figure 15). Instead of the Regex field and edge checklist (because having a checklist for arcs makes little sense), the tab has:

- **Order By** option with radio buttons for *Source* and *Target*. This allows the user to order the arcs in report by their source node name or target node name, respectively. By default, they are *ordered by source*.
- **Group By** option with radio buttons for *Place to Transition* and *Transition to Place*. If the user selects 'Place to Transition' grouping, then all the arcs directed from places to transitions are ordered and listed first in the table; remaining arcs (belonging to the Transition to Place group) follow. Similarly, if the other option is selected, arcs for this group precede the other edges in the list. By default, arcs are *grouped by Place to Transition*.

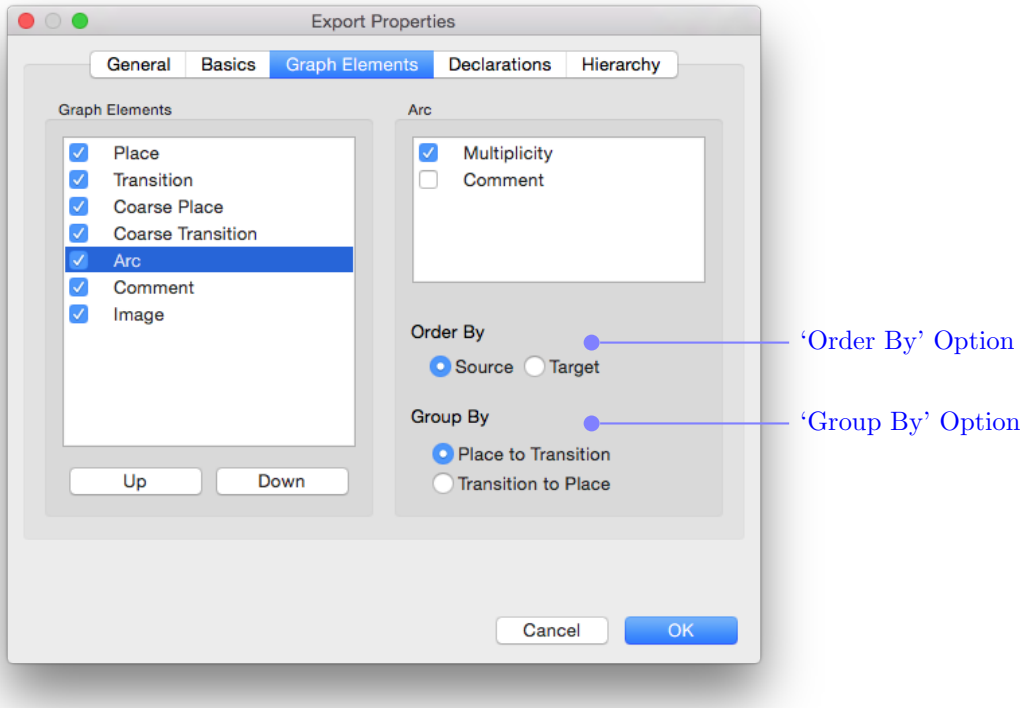


Figure 15: Template Design: Graph Elements tab for Edges

For a detailed utility description of all these elements, kindly refer to section A.4.2.3 of *User Manual*.

6.1.4 Declarations Tab

This is the fourth tab in the dialog (as illustrated in Figure 16). It lists the declarations for the given net and custom attributes for each declaration. Every net class has a specific set of declarations. Hence, the contents of this tab are also *dynamic to the net class for the given net.*

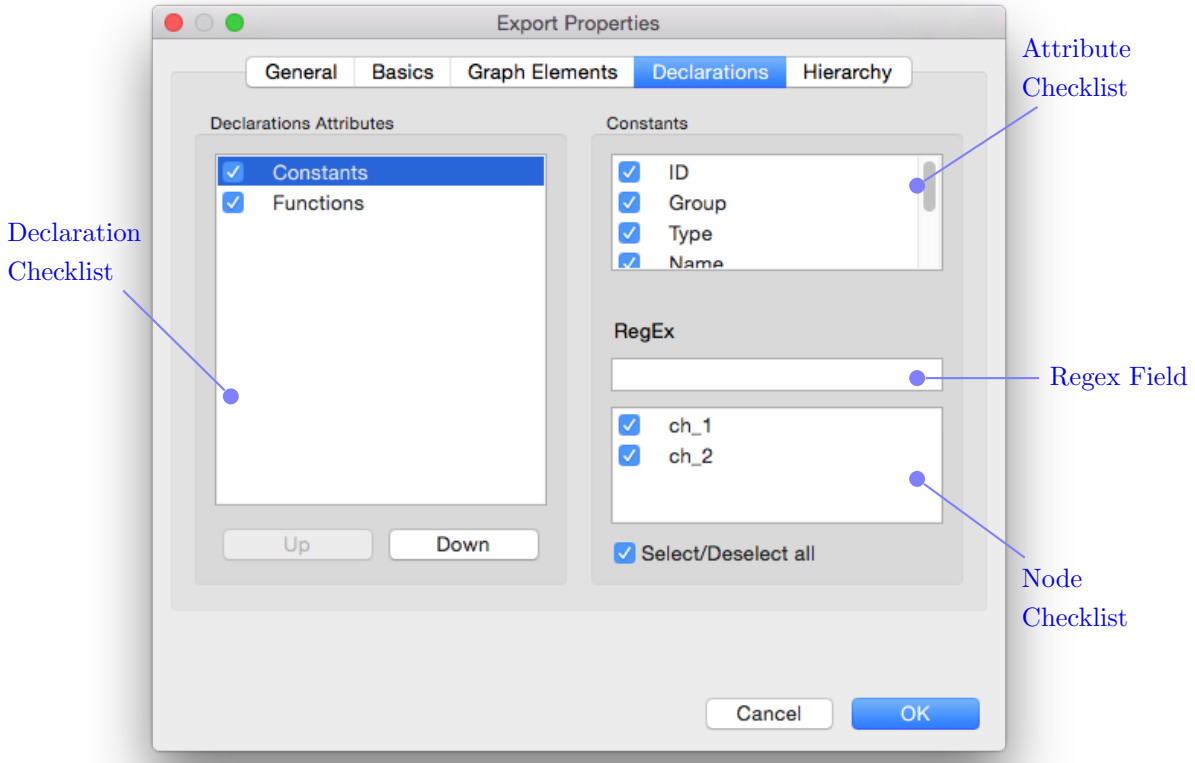


Figure 16: Template Design: **Declarations Tab.**

The Declarations tab also, like the Graph Elements tab, is organized into two panels.

- The **left panel** is a *rearrange panel*, as in Graph Elements tab. Thus, elements can be enabled/disabled or even reordered. The utility of this selection and ordering is explained in section 6.2. This panel lists the declarations for the given net.
- The **right panel** is *dynamic* to the current selection of declaration in the left panel. It lists the attributes and nodes corresponding to the selected declaration, for customization.

The generic structure of this tab is categorized into four main parts:

1. **Declarations Checklist**

   Each net class has a specific set of declarations. The left panel contains a checklist of declarations for the given net, which can be individually selected/deselected or reordered. For example, in Figure 16, the declarations for a net belonging to Petri net class are listed.

2. **Attribute Checklist**

   Every declaration element has a unique set of attributes. This top right panel contains the attributes for the current selection of declaration, as a checklist. Any attribute can thus be selected/deselected for export by checking/unchecking the respective item in the list. For example, in Figure 16, the attributes for 'Constants' are listed.

3. **Node Checklist**

   Every net has a unique set of nodes corresponding to each declaration. This bottom right panel contains the nodes for the current selection of declaration, as a checklist. Any attribute can thus be selected/deselected for export by checking/unchecking the respective item in the list. Besides, a **Select/deselect all** checkbox is provided to select/deselect the complete node list for export. For example, in Figure 16, the nodes of 'Constants' declaration type for the given net are listed.

4. **Regex Field**

   This **RegEx** field works similar to the regex field of *Graph Elements* tab, for its respective node list.

For a detailed utility description of all these elements, kindly refer to section A.4.2.4 of *User Manual*.

6.1.5 Hierarchy Tab

This is the last tab in the dialog (as illustrated in Figure 17). It corresponds to the hierarchical structure of the given net. It lists the hierarchy levels and custom attributes for the given net. Since every net has a specific hierarchical structure, this tab is *dynamic to the hierarchy for the given net.*

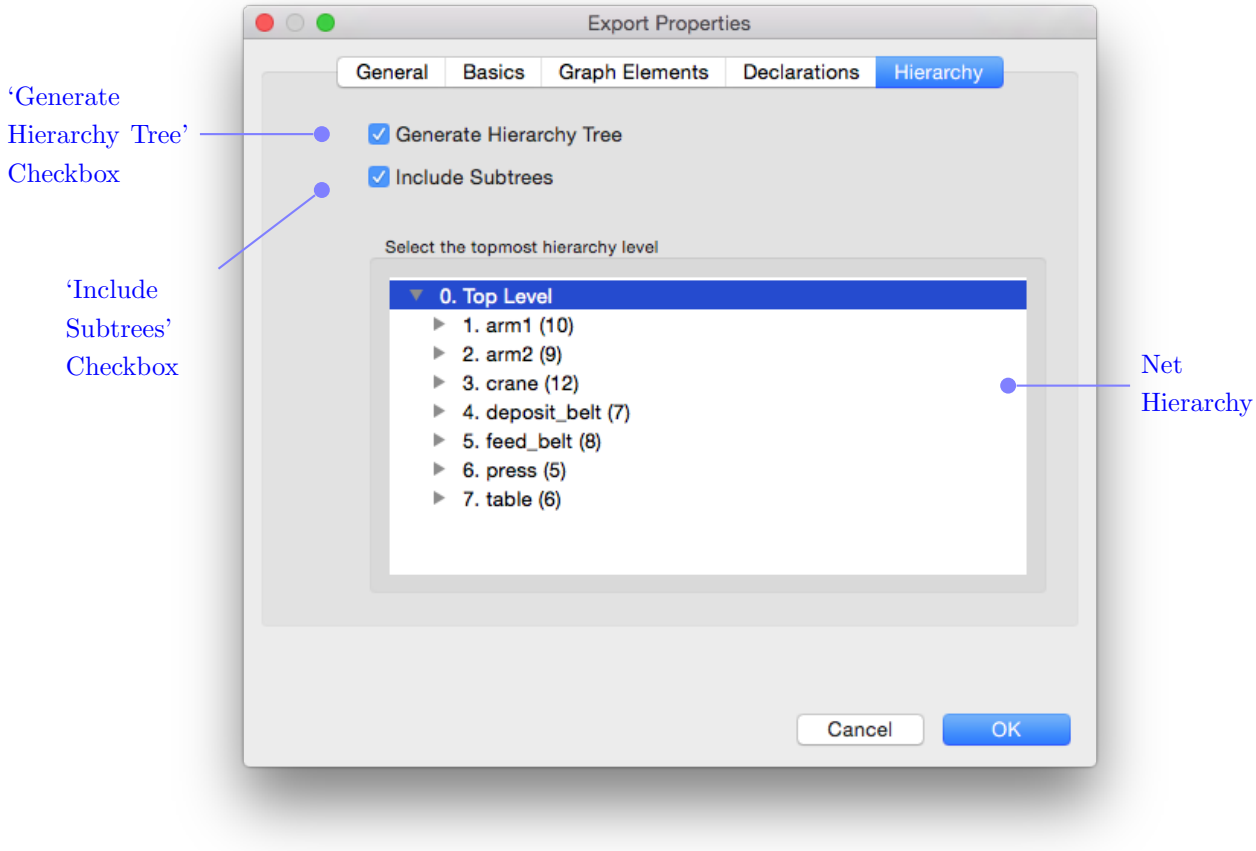


Figure 17: Template Design: **Hierarchy Tab**.

This layout of this tab is unlike that of *Basics, Graph Elements* or *Declarations* tabs. It needs to list the hierarchy of the net and the corresponding attributes for customization. Therefore, the generic structure of this tab is categorized into three main parts:

1. **Net Hierarchy**

   This panel shows the hierarchical structure for the given net, as displayed in the Hierarchy pane of Snoopy home window (Figure A.2). Besides, each net level label is prefixed with a net level enumeration for a more detailed hierarchical view. Further, this panel is a *multi-selection panel*, i.e., it allows a range of items to be selected. Therefore, the user can select single or multiple levels from this hierarchy to export the net figures corresponding to these levels. To export all subnets in the net hierarchy, **Top Level** can be selected.

   To avoid ambiguity, if a level and one or more of its sublevels are simultaneously selected for export, then only the common ancestor node to all such levels is considered for export.

2. **'Generate Hierarchy Tree' Checkbox**

   This option is added to provide an additional detailing about the net hierarchy in the generated report. If selected, it shall render a tree like structure for each selection of hierarchy level, thus graphically illustrating the hierarchical positioning of the corresponding net.

3. **'Include Subtrees' Checkbox**

   This option shall provide an additional user-control over the exported hierarchy levels. If selected, then corresponding to each selection of hierarchy level, all its sub-levels (whether expanded/collapsed in the 'Net Hierarchy' panel) are also exported to the report (and are included in the hierarchy tree for the level, if 'Generate Hierarchy Tree' option is selected).

   If this option is unchecked, only the immediate selections for hierarchy levels and the corresponding sub-levels expanded in the 'Net Hierarchy' panel are exported.

For a detailed utility description of all these elements, kindly refer to section A.4.2.5 of *User Manual*.

6.2 Template Design for Generated Report

Template for the generated report is designed in accordance with Snoopy2LaTeX Export dialog (detailed in the preceding section 6.1). The main features of this report structure are:

- **Modular Structure**

  The generated LaTeX code is split across several interlinked files rather than a single large file for one net. Each file contains information pertaining to a specific export entity. This makes the code more **readable** and easy to **reuse**, **edit** and **debug**.

- **Standard format**

  The generated report follows a standard format, and its contents can thus be entirely or partly used for scientific documentation directly.

- **Structure analogous to Export dialog**

  The report is structured such that the positioning of each export element in the report can be mapped to its corresponding custom positioning in the Export dialog. Therefore, user can easily related between the exported net and its generated report.

- **Extensive cross-referencing**

  The generated report consists of efficient cross-referencing for all its elements. Each element label is mapped to its respective detail in the report as a hyper-reference.

The detailed structure of the report template is elaborated in the following subsections. For a detailed utility description of the generated report, kindly refer to section A.5 of *User Manual*.

6.2.1 Basic Structure of Exported LaTeX Code

The generated LaTeX code is structured as a collection of several interlinked LaTeX files. Each file contains exported code for a specific export entity in the Export dialog.

The basic template for generate code follows a standard format, with a structural flow as:

1. Title Page
2. Table of Contents
3. Sections specific to net information
4. References
5. Glossary

The template for LaTeX code generated can be given as:

```
\documentclass[pdftex,12pt]{article}
<preamble>

\begin{document}

\input{./TitlePage.tex}

\newpage
\tableofcontents

\input{./Basics.tex}
\input{./GraphElements.tex}
\input{./Declarations.tex}
\input{./Hierarchy.tex}
\input{./References.tex}
\input{./Glossary.tex}

\end{document}
```

As illustrated in the template above, each tab in the Export dialog corresponds to a separate LaTeX file which is included to the main TeX document. LaTeX code for each tab is further split into separate files, one for each element listed under that tab. Each such file is organized as a collection of sections and subsections.

- For the **Basics** as in Export dialog, the section for Basics is split into two subsections *General Informations* and *Net Informations*.
- For the **Graph Elements** and **Declarations** as in Export dialog, the respective sections are split into subsections, one for each element of that type. For example, a typical TeX file generated for 'Graph Elements' for a Petri net looks like this:

```
\newpage

\section{Graph Elements}
This section contains information related to graph elements specific
to the net.

\input{./Places.tex}
\input{./Transitions.tex}
\input{./CoarsePlaces.tex}
\input{./CoarseTransitions.tex}
\input{./Arcs.tex}
\input{./Comments.tex}
\input{./Images.tex}
```

- For **Hierarchy** as in Export dialog, the section is split into two subsections: *Hierarchy Tree* and *Hierarchy Figures*.
  - **Hierarchy Tree** subsection shall display tree-like structures to illustrate the hierarchical organization of the net levels selected for export.
  - **Hierarchy Figures** subsection shall display the actual net figures for the selected net levels.

Further, for each element belonging to *Graph Elements* or **Declarations**, the corresponding section contains information about the element against the selected attributes and nodelist as a **longtable** in LaTeX, where:

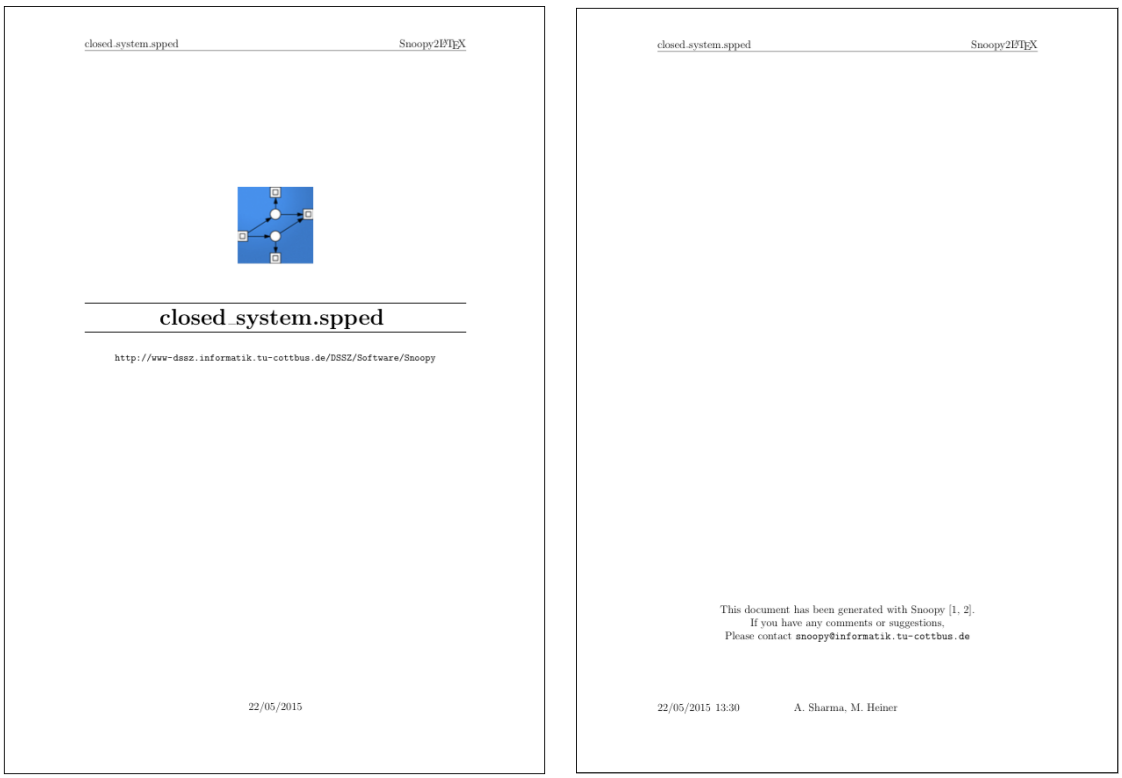
- **Nodes** selected for export form the rows of table, and
- **Attributes** selected for export form the columns of table

6.2.2 Basic Structure of Generated PDF

The PDF report is generated on compiling the exported LaTeX code. As defined in the basic template for generated LaTeX code, the PDF is structured as:

1. **Title Page (and Desclaimer)**

   The title page (as illustrated in Figure 18) shows the net name, URL for Snoopy website, current date as footer and 'Snoopy2LaTeX' as header. The next page contains a *desclaimer* for the Export. Every generated PDF starts with these two pages.

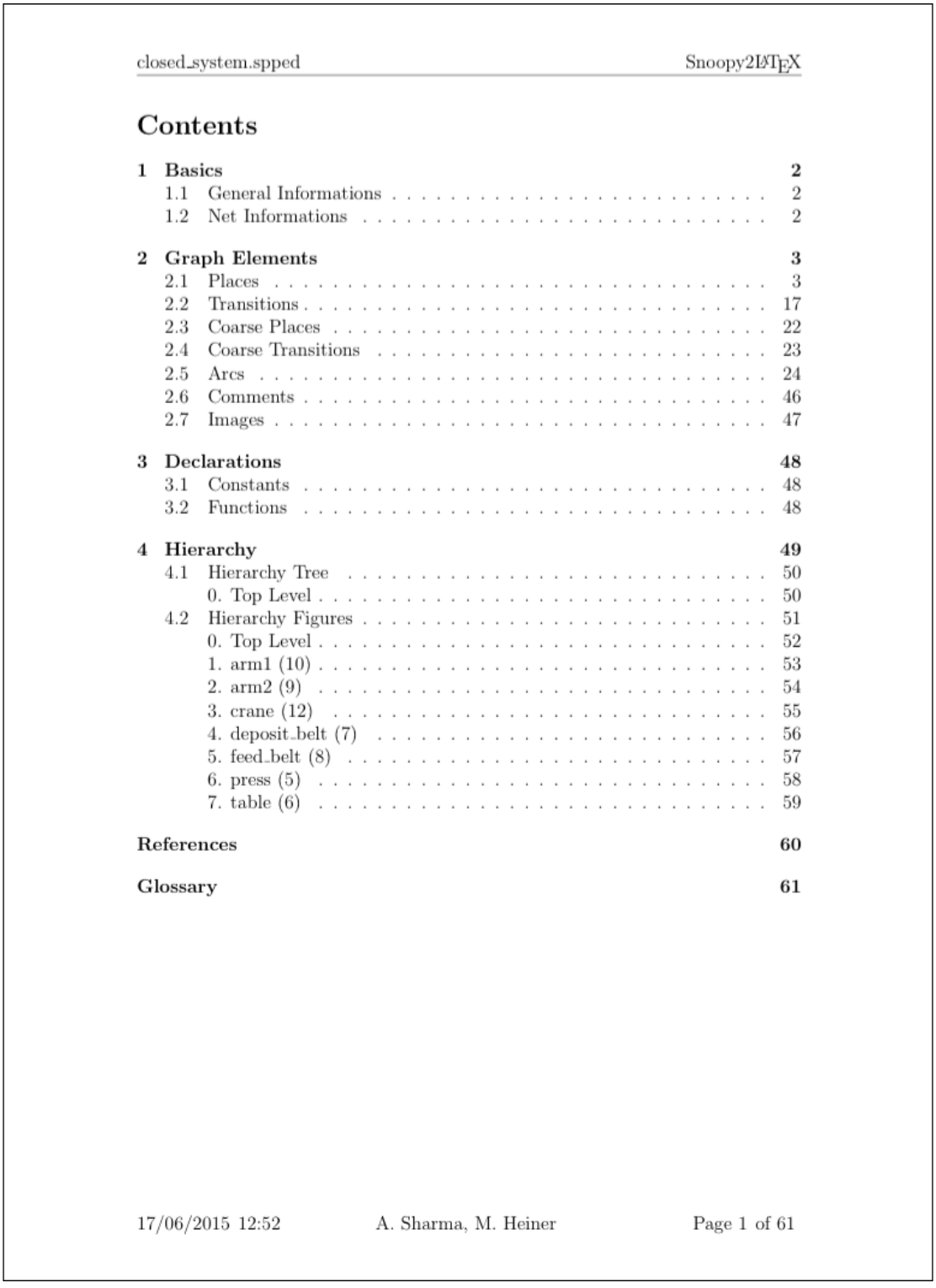


(a) Title Page (b) Desclaimer

Figure 18: Generated PDF: Title Page & Desclaimer

2. **Table of Contents**

The table of contents, as for a typical report, lists the sections and subsections of the report. In case of our Export to LaTeX, it clearly lists the elements exported for the given net. Figure 19 shows the table for contents for a generated PDF sample.



closed_system.spped Snoopy2LaTeX

Contents

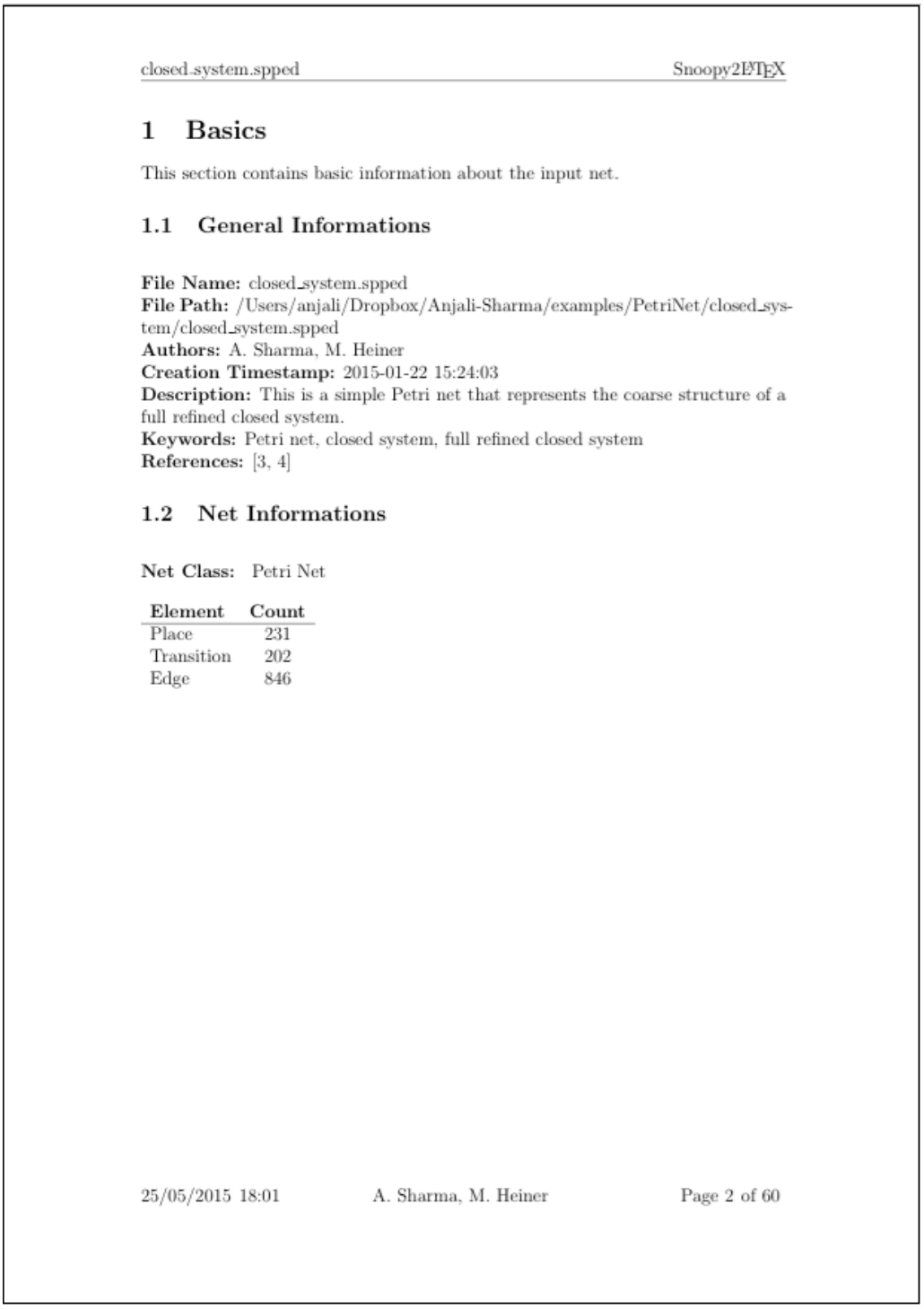
17/06/2015 12:52 A. Sharma, M. Heiner Page 1 of 61

Figure 19: Generated PDF: Table of Contents. It illustrates the contents of a generated report for a sample Petri net export.

3. **Sections specific to net information**

   This corresponds to the net-specific information. For each selected tab in the Export dialog (except, General tab), a section is generated in the PDF (as shown in Figure 19). The sections are arranged in the order as specified in the dialog.

   As defined in the previous section, the *Basics* section is split into two subsections: 'General Informations' and 'Net Informations' with the basic net information. Figure 20 shows the Basics section for a sample net.



closed_system.spped Snoopy2LaTeX

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: closed_system.spped
File Path: /Users/anjali/Dropbox/Anjali-Sharma/examples/PetriNet/closed_system/closed_system.spped
Authors: A. Sharma, M. Heiner
Creation Timestamp: 2015-01-22 15:24:03
Description: This is a simple Petri net that represents the coarse structure of a full refined closed system.
Keywords: Petri net, closed system, full refined closed system
References: [3, 4]

1.2 Net Informations

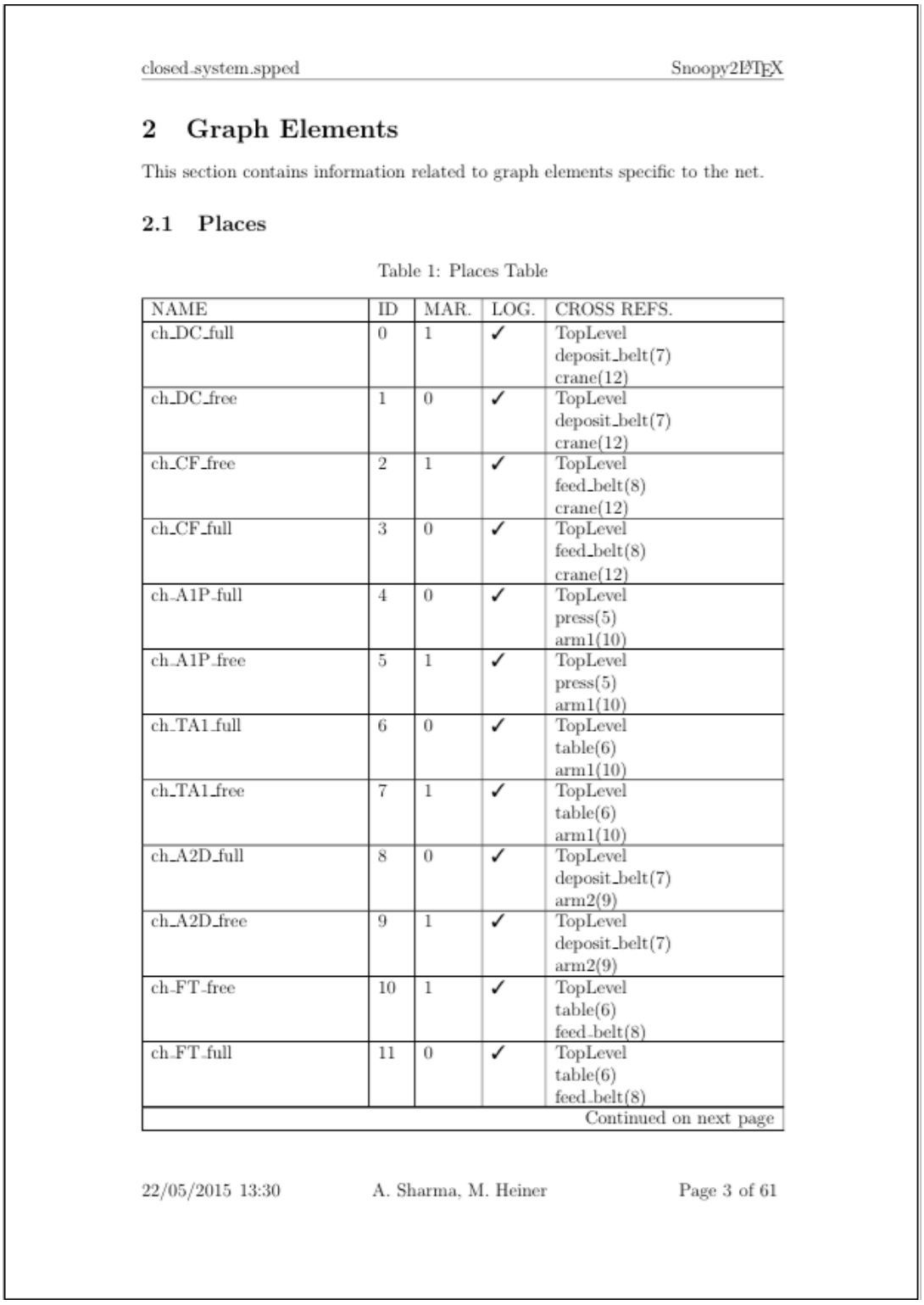
Net Class: Petri Net

Element	Count
Place	231
Transition	202
Edge	846

25/05/2015 18:01 A. Sharma, M. Heiner Page 2 of 60

Figure 20: Generated PDF: Basics for Sample Net

As defined in the previous section, each element belonging to *Graph Elements* or *Declarations* is exported as a table with nodes as rows and attributes as columns. Figure 21 shows the table generated for selected 'Places' of an example Petri net.



closed_system.spped Snoopy2LaTeX

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ch_DC_full	0	1	✓	TopLevel deposit_belt(7) crane(12)
ch_DC_free	1	0	✓	TopLevel deposit_belt(7) crane(12)
ch_CF_free	2	1	✓	TopLevel feed_belt(8) crane(12)
ch_CF_full	3	0	✓	TopLevel feed_belt(8) crane(12)
ch_A1P_full	4	0	✓	TopLevel press(5) arm1(10)
ch_A1P_free	5	1	✓	TopLevel press(5) arm1(10)
ch_TA1_full	6	0	✓	TopLevel table(6) arm1(10)
ch_TA1_free	7	1	✓	TopLevel table(6) arm1(10)
ch_A2D_full	8	0	✓	TopLevel deposit_belt(7) arm2(9)
ch_A2D_free	9	1	✓	TopLevel deposit_belt(7) arm2(9)
ch_FT_free	10	1	✓	TopLevel table(6) feed_belt(8)
ch_FT_full	11	0	✓	TopLevel table(6) feed_belt(8)

Continued on next page

22/05/2015 13:30 A. Sharma, M. Heiner Page 3 of 61

Figure 21: Generated PDF: Sample Table for Places.

4. **References**

   This section contains the references for the net (illustrated in Figure 22). It includes two basic references for Snoopy which are by default included for every generated report. Besides, it includes references for the given net as specified in *General Informations* dialog of Snoopy.

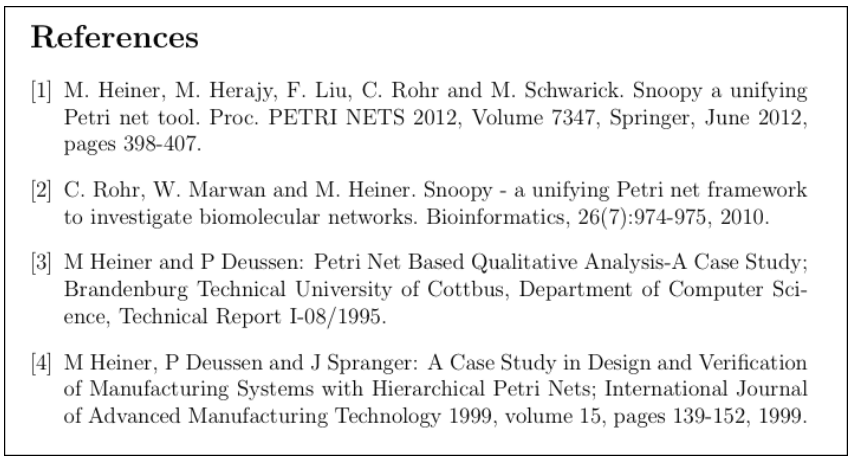


Figure 22: Generated PDF: References for a Sample Net

5. **Glossary**

   This is the last section of the report and contains a glossary of the basic abbreviations used in the report (illustrated in Figure 23).

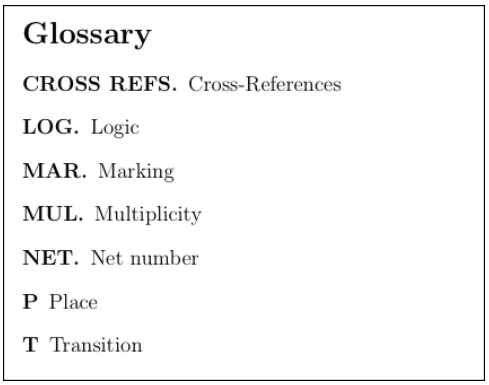


Figure 23: Generated PDF: Glossary for a Sample Net

6.2.3 Structure of Hierarchy Export

The *Hierarchy* section is discussed separately as it is structured unlike the other tabs. This section is has two subsections: *Hierarchy Tree* and *Hierarchy Figures* as defined in section 6.2.1.

- **Hierarchy Tree**

  This subsection shows a tree-like structure for each net level selected for export. This tree illustrates the hierarchical position of the corresponding net levels in the overall hierarchy for the given net. Figure 24 shows hierarchy tree for one selected level of a sample net.

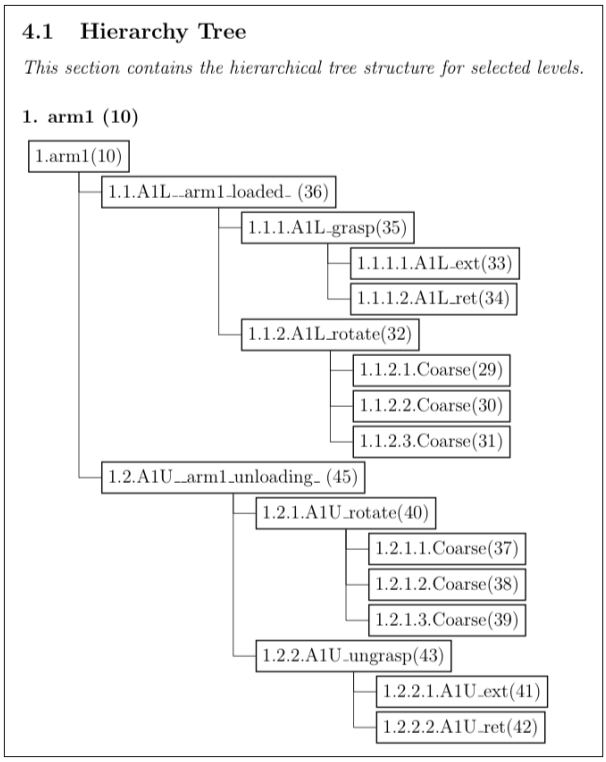


Figure 24: Generated PDF: Hierarchy Tree for a Sample Net. It shows the hierarchy tree for a level selected for export. The subnets are organized as nodes of the tree according to the net hierarchy.

- **Hierarchy Figures**

  This subsection contains net figures for all the levels selected for export. Figure 25 shows the net figure as exported for the top level of a sample net.

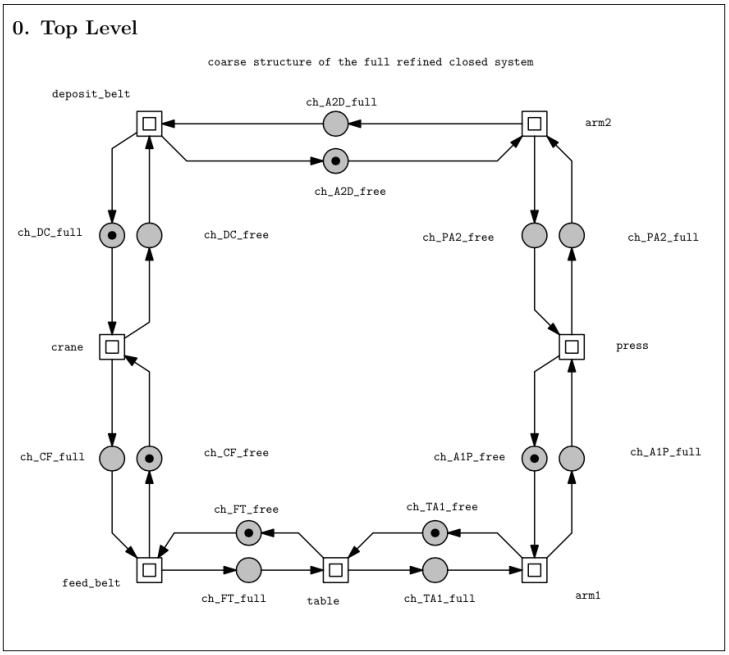
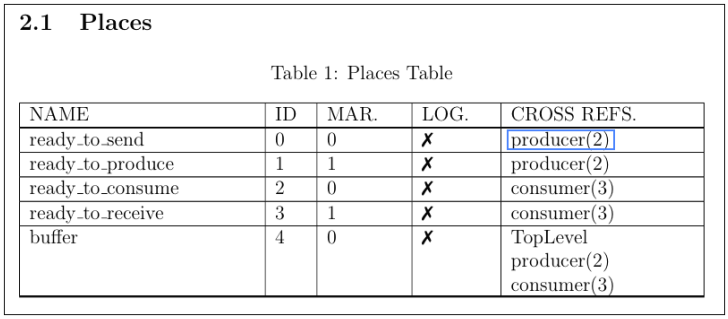


Figure 25: Generated PDF: Hierarchy Figure for a Sample Net. It shows the hierarchy figure for a level selected for export. The subnets are organized as figures, one for each net level.

6.2.4 Cross-referencing Elements

For large nets with hundreds and thousands of nodes, the generated report can get significantly long. This makes it difficult to search a specific row of a specific table in the report describing a certain node of the net. Therefore, an additional feature of **Cross-referencing** is introduced to each element of *Graph Elements*, *Declarations* and *Hierarchy* tab.

- For each element of *Graph Elements* or *Declarations* tab, if selected, this attribute renders a new column **CROSS REFS.** to the corresponding element table. For a given node entry in the table, it lists all the subnets where that node is used. Each entry of this list is a *hyper-reference* to the corresponding net figure in the *Hierarchy Figures* section of the report. Figure 26 illustrates an example scenario.

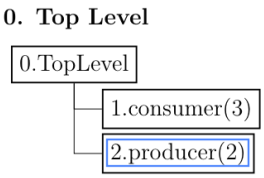


2.1 Places

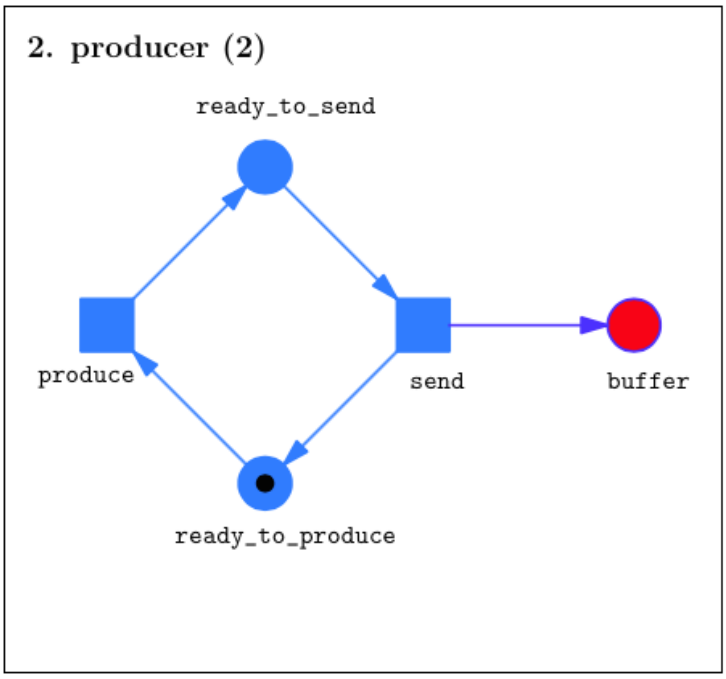
Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ready_to_send	0	0	✗	producer(2)
ready_to_produce	1	1	✗	producer(2)
ready_to_consume	2	0	✗	consumer(3)
ready_to_receive	3	1	✗	consumer(3)
buffer	4	0	✗	TopLevel producer(2) consumer(3)

(a) Places Table in Report



(b) Hierarchy Tree in Report



(c) Net figure for a given net level

Figure 26: Generated PDF: Cross-reference. In Figure (a) and (b), the level label enclosed in a blue rectangle indicates a cross-reference. This when clicked, leads to the corresponding net figure as shown in Figure (c). All net levels have similar cross-referencing in the generated report.

- Each node name in **Arcs Table** links to the tuple with its description. For instance, if the node is a Coarse Place, then it links to the tuple for this node in the 'Coarse Places Table'.
- Each node in the **Hierarchy Tree** also links to its corresponding figure in the 'Hierarchy Figures' subsection. Figure 26 illustrates an example scenario.
- If **Images** are present in a net and are exported, then thumbnail for each exported image in the **Image Table** links to its enlarged version in the appendix.
- By the default property of LaTeX, every every in the table of contents for the report links to its corresponding section.

7 Implementation

The new Export to LaTeX is coded as an extension of the existing implementation. It was carried out in two steps. Firstly, the new design for the Snoopy2LaTeX Export dialog was implemented. Next, the LaTeX code to be generated during export was worked on.

The following subsections explain these implementations in detail.

7.1 Export Dialog UI Implementation

Snoopy is a platform-independent tool and the new Export functionality must comply with this feature. Thus, Export dialog user interface (design explained in section 6.1) is implemented using the *Wxwidgets - a C++ cross-platform GUI library* [11].

- A major feature expected from Snoopy2LaTeX is that all the tabs and their elements can be *rearranged*. To do this, the list of tab labels and the element list in each tab is implemented using **wxRearrangeList** which allows the items to be selected/deselected as well as be reordered using 'Up' and 'Down' buttons.
- This export is expected to be *generic to all net classes*. This is facilitated by a **generic implementation**. Each tab is implemented by a separate function:
  - ***AddGeneral()*** adds the General tab and its contents. It has a static behaviour independent of the net class for the given net.
  - ***AddBasics()*** adds the Basics tab to the dialog. The contents of this tab are also independent of the net class. Functions *AddAttributes_BasicsGeneral(), AddAttributes_BasicsNet(), AddAttributes_BasicsLayout()* and *AddAttributes_BasicsTypography()* add contents to the right panel of Basics tab for General Informations, Net Informations, Report Layout and Report Typography, respectively.
  - ***AddGraphElements()*** adds the Graph Elements tab to the dialog. The graph elements are dynamic to the net class. Hence, depending on the net class, this function loads the corresponding elements to the left panel. The nodes and attributes are added to the right panel by *AddAttributes_GraphElements()* function.
  - ***AddDeclarations()*** adds the Declarations tab to the dialog. The declarations are also dynamic to the net class. Hence, depending on the net class, this function loads the corresponding elements to the left panel. The nodes and attributes are added to the right panel by *AddAttributes_Declarations()* function.
  - ***AddHierarchy()*** adds the Hierarchy tab to the dialog. Helper function *copyTree_recur()* recursively adds all hierarchy levels to the tree control in this tab.
- Function ***LoadData()*** fetches all the nodes and attributes for the given net as soon as *AddToDialog()* gets called by the parent class. These fetches values are then loaded to the respective node and attribute checklists in the dialog by the functions listed above.
- **Dynamic event handling** is used for each UI element of the Export dialog.

7.2 Latex Code Implementation

After the implementation of a complete working UI for the Export dialog, the LaTeX code to be generated as output and the functionality for its compilation was implemented.

- Firstly, the virtually inherited ***Write()*** function was modified to invoke function *StartDoc()* to start a new document, *WriteLatex()* to write the LaTeX code into the document, and *EndDoc()* to end it.
- ***WriteLatex()*** function firstly writes the code for title page using *WriteTitlePage()*, defines the page layout and then adds the code for tab contents as per their order defined in the dialog.
  - ***WriteBasics()*** writes the LaTeX code corresponding to the elements of Basics tab.
  - ***WriteGraphElements()*** writes the LaTeX code corresponding to the elements of the Graph Elements tab.
  - ***WriteDeclarations()*** and ***WriteDeclarations_Colored()*** functions write the LaTeX code for the elements in the Declarations tab for uncolored and colored nets, respectively (Declarations are handled separately for colored and uncolored nets due to the difference in their structure).
  - ***WriteHierarchy()*** writes the code corresponding to the Hierarchy tab. It calls *WriteHierarchyTree()* function which writes the hierarchy trees for the selected levels, and *WriteHierarchyFigures()* which writes the hierarchy figures (using *tikz* library of LaTeX) for the selected levels.
  - ***WriteReferences()*** and ***WriteGlossary()*** functions write the references as *bibitems* and glossary for the report, respectively.

The *Write()* function is called by the parent class as soon as the 'OK' button of the dialog is pressed, and the LaTeX code is then generated, as desired, using *WriteLatex()*.

8 Testing

After one round of complete implementation, iterative testing and improvement of the Export functionality was done.

8.1 Testing on Different Net Classes

Testing of Snoopy2LaTeX Export for different net classes was carried in four phases:

1. **Testing for Qualitative Nets**

   This includes testing of Snoopy2LaTeX Export for several simple as well as complex nets for each of the following net classes:

   (a) Petri Net
   (b) Extended Petri Net
   (c) Reachability Graph
   (d) *Music Petri Net (pending...)*

2. **Testing for Quantitative Nets**

   This includes testing of the Export for several simple and complex nets (with and without rate functions) for each of the following net classes:

   (a) Stochastic Petri Net
   (b) Continuous Petri Net
   (c) Hybrid Petri Net
   (d) Time Petri Net

3. **Testing for Colored Nets**

   This phase involves testing for all the colored variants of net classes which include:

   (a) Colored Petri Net
   (b) Colored Stochastic Petri Net
   (c) Colored Continuous Petri Net
   (d) Colored Extended Petri Net
   (e) Colored Hybrid Petri Net

4. **Testing for Other Nets**

   This phase involves testing for all other net classes which include:

   (a) Freestyle Net
   (b) Modulo Petri Net
   (c) MTBDD
   (d) MTIDD
   (e) *Fault Trees (pending...)*
   (f) *Extended Fault Trees (pending...)*

Sample reports generated for example nets (one for each net class) are demonstarted in section A.7 of *User Manual*.

8.2 Testing on Different OS Platforms

Snoopy2LaTeX Export, like Snoopy, is implemented to be platform-independent. To ensure smooth functioning of this export on different platforms, testing was carried out for several simple and complex nets on OS platforms:

1. Mac OS X 10.10
2. Linux: Ubuntu 12.10
3. Windows 7

9 Snoopy2LaTeX vs. SBML2LaTeX: Comparison

Snoopy2LaTeX and SBML2LaTeX are two independent export tools that extensively utilize the potential of LaTeX to generate high-quality scientific documentation for their respective inputs. The initial template design for Snoopy2LaTeX is heavily inspired by SBML2LaTeX. However, some additional features are introduced to the tool as well.

Here we discuss the major similarities and differences among these two exports.

9.1 Similarities

The major similarities between the two exports or to say the features of Snoopy2LaTeX inspired by SBML2LaTeX are:

- **Layout customization options**

  Both the export tools offer several options for customization of the layout of generated LaTeX report. This include custom options for font size, font styles, paper orientation, addition of title page to report etc.

  Figure 27 shows the Export dialog for SBML2LaTeX. Note the *Layout options* and *Typography options* provided. In Snoopy2LaTeX, similar layout options (and some more) are provided under the *Report Layout* and *Report Typography* elements of the **Basics** tab, as illustrated in Figure 12 and 13, respectively.

- **Content customization options**

  Besides layout, these export tools provide custom options for even the content of the generated report. SBML2LaTeX offers content customization as a set of generic checkbox options (illustrated in *Report options* and *Content of the report* sections in Figure 27). Snoopy2LaTeX extends this functionality by providing custom options for each export element, dynamic to the current input net. For example, Figure 14 shows the custom options for *Places* in the input net, while Figure 17 shows custom options for its hierarchy.

- **Information arranged into tables**

  The generated textual content is organized into clearly arranged tables, thus simplifying the task of understanding and communicating the model as well as detecting and correcting errors.

- **Direct export to human-readable formats**

  Both the tools allow the compilation of exported LaTeX code from within the Export dialog and direct generation of the report in human-readable formats. While SBML2LaTeX supports file formats PDF, TeX, DVI and PS; Snoopy2LaTeX allows generation for a PDF report.

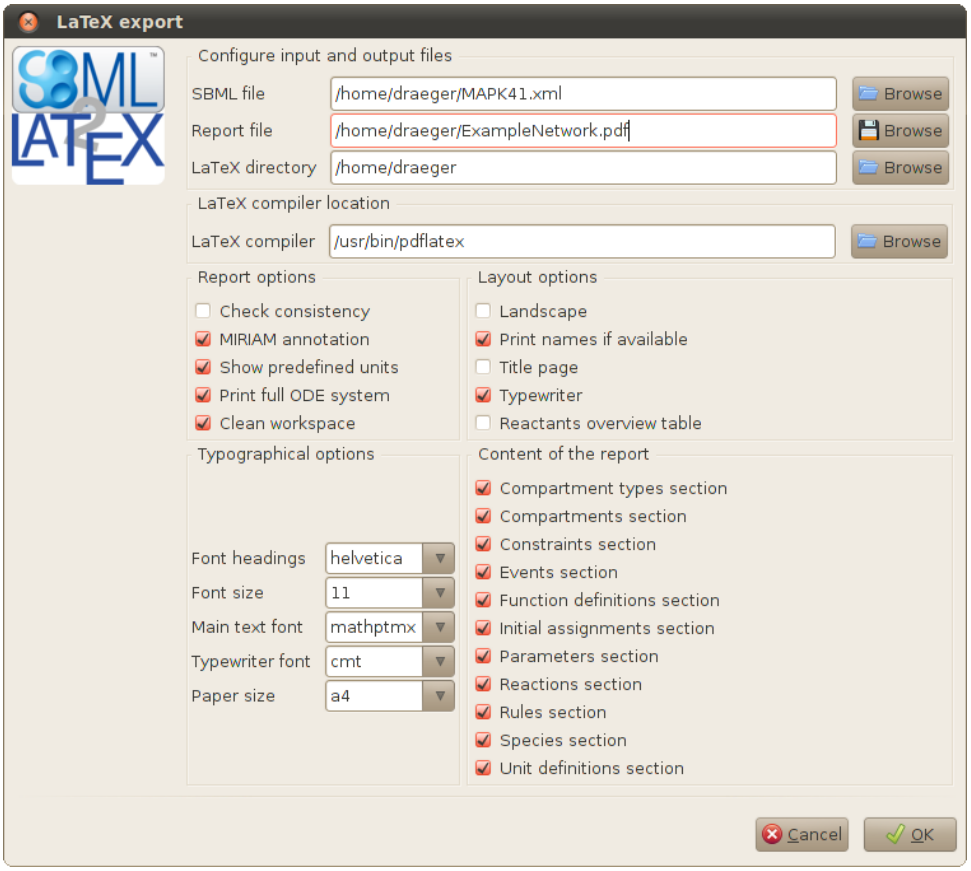


Figure 27: SBML2LaTeX Export Dialog. This tool, as shown, provides several options to customize the layout, typography as well as content of the output report. Is is used as a basis for the dynamic and static options implemented in the new Snoopy2LaTeX Export. Taken from [8].

9.2 Differences

The differences here correspond to the additional functionalities implemented in Snoopy2LaTeX that are not supported by SBML2LaTeX

- **Customization of each export element individually**

  While both the export tools provide options to customize the report content, Snoopy2LaTeX makes these options more specific and dynamic to the contents of input net. For the input net, each graph element, declaration and hierarchy level can be individually customized in this Export. SBML2LaTeX, on the other hand, provides a fixed set of options as checklist which are independent of the contents of input net.

- **Reordering of export elements**

  As SBML does not contain mandatory components, SBML2LaTeX displays the sections about the following components in this order if they are declared in the model: compartment types, compartments, species types, species, global parameters, initial assignments, function definitions, rules, events, constraints and reactions [3].

  Snoopy2LaTeX, unlike this, allows each export element to be reordered in the Export dialog. The elements are then exported to LaTeX in this new order. Besides, all the nodes for the input with a *name* attribute can be ordered lexicographically.

- **Cross-referencing in generated report**

  Snoopy2LaTeX provides extensive cross-referencing among the exported elements. Each reference to a node in the report links to its description in the corresponding element table. Also, every reference to a hierarchy level links to its net figure in the report.

  SBML2LaTeX does not provide this feature and a simple LaTeX report is generated with no cross-referencing.

- **Modular structure of generated report**

  Snoopy2LaTeX generates the output LaTeX code as a collection of interlinked TeX files. Each files contains information for a specific export element. This makes the report easy to *edit, reuse* and *debug*.

  SBML2LaTeX generates the report as a single TeX file which can get significantly long for large nets, and therefore difficult to edit or debug.

10 Summary

10.1 Achievements

1. An extensively enhanced version of Snoopy2LaTeX has been successfully implemented, that increases the utility of this Export manifold. The major features include:
   (a) Export to LaTeX for not just the net figure (as done by Export to LaTeX in current Snoopy version) but also the graph elements, declarations and hierarchy of the net. Even the metadata of the net can be exported to LaTeX now.
   (b) Generation of a complete standard report for a given net, contents of which can be directly used for scientific documentation.
   (c) Complete user control over the layout and content of the generated report. Each export element can be individually customized for export.
   (d) Option for direct generation of PDF report using Snoopy2LaTeX Export. This makes generation of detailed documentation for a given Snoopy net, just clicks away.
2. The implementation for generated code as output required extensive knowledge of LaTeX. Thus, during the development phase, a significant understanding of LaTeX was developed. This report is written in LaTeX as well.
3. The UI is implemented using C++ cross platform GUI library - wxWidgets, which rendered a deep understanding of cross-platform software development.

10.2 Open Problems

There are few potential areas where this work can be extended or improved.

1. **Export for Fault Trees** is not functional due to some incompatibilities with their implementation.
2. **Automatic orientation of pages** with wide tables to *landscape* mode, in the generated report.
3. **Resolution of mathematical expressions exceeding column width** in the element tables of generated report due to constraints of hyphenation in math mode of LaTeX.
4. **Support for more TeX compilers** can be added. Currently, only *pdfatex* and *latexmk* are used to generate PDF report.
5. **Auto-detection of TeX compiler path** in user system is a significant feature that can be implemented.
6. Some **issues pertaining to LaTeX capabilities** are faced. These include:
   (a) Over lapping text and horizontal lines for some tables in the generated report
   (b) Blank pages in between Hierarchy figures in generated PDF report
   (c) Mathematical expressions like rate functions overflowing the table cells in report due to no facility of automatic hyphenation in math mode of LaTeX

References

[1] F. Liu and M. Heiner, *Petri Nets for Modeling and Analyzing Biochemical Reaction Networks*, chapter 9, pages 245–272. Springer, 2014.

[2] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick, *Snoopy a unifying Petri net tool*, Proc. PETRI NETS 2012, Volume 7347, pages 398–407, Springer, June 2012.

[3] A. Drager, H. Planatscher, D.M. Wouamba, A. Schroder, M. Hucka, L. Endler, M. Golebiewski, W.Muller and Andreas Zell, *SBML2LATEX: Conversion of SBML files into human-readable reports*, In: Advance Access publication, Volume 25 no. 11, pages 1455–1456, March 23, 2009.

[4] W. Marwan, C. Rohr, and M. Heiner, *Petri nets in Snoopy: A unifying framework for the graphical display, computational modelling, and simulation of bacterial regulatory networks*, Volume 804 of Methods in Molecular Biology, chapter 21, pages 409–437. Humana Press, 2012.

[5] M. Heiner, D. Gilbert and R. Donaldson, *Petri Nets for Systems and Synthetic Biology*, In SFM 2008, (M Bernardo, P Degano and G Zavattaro, Eds.), Springer, pages 215–264, 2008.

[6] C Rohr, W Marwan and M Heiner. *Snoopy - a unifying Perti net framework to investigate biomolecular networks*, Bioinformatics, 26(7):974–975, 2010.

[7] Petri Net, `http://en.wikipedia.org/wiki/Petri_net`
(Last accessed: March 25, 2015)

[8] SBML2LaTeX, `http://www.ra.cs.uni-tuebingen.de/software/SBML2LaTeX/`
(Last accessed: February 15, 2015)

[9] Latex, `http://en.wikipedia.org/wiki/LaTeX`
(Last accessed: March 30, 2015)

[10] LaTeX - A document preparation system, `http://www.latex-project.org`
(Last accessed: March 30, 2015)

[11] Wxwidgets, `http://www.wxwidgets.org`
(Last accessed: May 20, 2015)

Appendices

A Snoopy2LaTeX: User Manual Version 1.0

A.1 Introduction

Snoopy is an efficient tool for modeling biochemical reaction networks and simulating them. It supports import and export functionality for various file formats to render executable results for a wide range of platforms, significant for any typical research work. On the other hand, LaTeX is a document markup language that serves as the *de facto* standard for the communication and publication of scientific documents. Therefore, **Export to LaTeX** functionality of Snoopy emerges out as an extensive feature with a wide range significance.

In this manual we discuss the use of Snoopy2LaTeX Export, that facilitates exporting Snoopy files to corresponding LaTeX code and further to human-readable PDF reports. There a several functionalities provided by Snoopy2LaTeX. If you are wondering about what this Export feature can add to your work-flow, it might be worth reading the following paragraphs.

Here are some aspects of what you can do using Snoopy2LaTeX:

- **Export Graph elements, Declarations and Hierarchy**

  All the graph elements, declarations and hierarchical elements of the given net can be efficiently exported to corresponding LaTeX code as a collection of tables; one for each node/edge/metadata class rendering complete (or as customized) information about all nodes belonging to that class.

- **Export Metadata of the net**

  Besides the net elements, even the metadata of the net that includes information such as the element count, net creation timestamp, authors, description etc. can be exported to LaTeX code with just a few clicks through the corresponding checklist in the Export dialog.

- **Generate Hierarchy Tree for the net**

  An additional feature that users can utilize is the option to generate a complete hierarchy tree or multiple tree structures corresponding to selected hierarchy levels. This functionality aims to render them a quick view through the organization of hierarchy levels across the net, which can be extremely useful in case of nets with complex hierarchical structure.

- **Extensively customize each export element**

  Users have the option to customize the list of attributes and elements they wish to export corresponding to each node/edge/metadata class. They can select a subset of hierarchy levels, metadata or even the complete element classes (Graph Elements, Declarations and Hierarchy) to be exported.

- **Control the layout and typography of the generated report**

  Detailed layout and typography options are provided to allow the users to have a control over the exact format and style of the generated report. User-defined header and footer can be inserted to the report, besides timestamp and page numbers. Orientation can be switched between portrait and landscape mode as well.

- **Reorder almost everything**

  Almost everything that can be exported can be reordered, as desired in the generated report. The sections corresponding to the tabs of the dialog and subsections corresponding to the element classes can be efficiently reordered by simply rearranging them in Export dialog using 'Up' and 'Down' buttons. Besides, the elements belonging to each element class can be ordered lexicographically, if they have a *name* attribute.

- **Use regular expressions for string selection**

  Users have the option to select a subset of elements to export using regular expressions. Besides, they can select/deselect the entire element list with a single click.

- **Generate PDF directly from the Export dialog**

  Besides exporting to LaTeX, users have the option to directly compile the generated LaTeX files and retrieve the resultant PDF report. They can even chose between TeX compiler chains *pdflatex* and *latexmk* to be used for compilation.

- **Generate a standard report using the default settings**

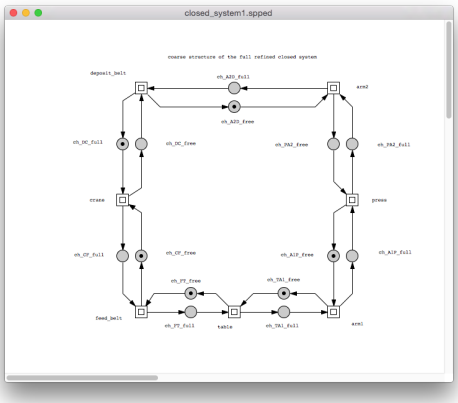
  While users can extensively customize each element, they also have the option to generate a report with default settings. The default settings provided for this Export render a PDF report (and corresponding LaTeX files) containing a detailed analysis for the given net.

- **Platform-independent implementation**

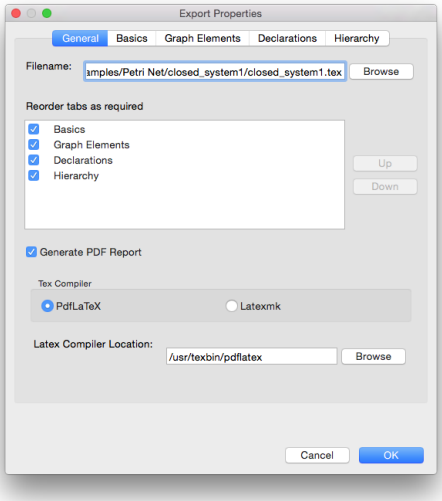
  The tool is implemented in standard C++, including the user interface that is implemented using wxWidgets 3.0.2 - C++ Cross-platform GUI Library. Hence, it can be efficiently used on different platforms which include Windows, Mac OS X, and Linux.

A.2 Application Scenario

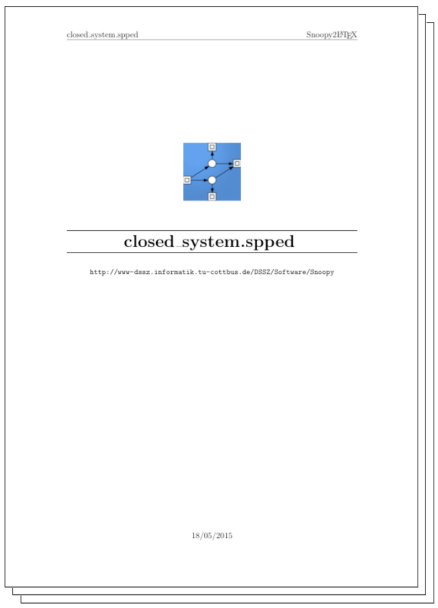
In a typical application scenario, the user constructs a biochemical reaction network using the Petri net modelling tool, Snoopy. Later, in order to export the constructed net to LaTeX, the user selects 'Export to LaTeX' option from the Export dialog. The resulting dialog facilitates user to customize the various elements of the net for the report, reorder them, select/deselect generation of PDF etc. After the customization of the report or proceeding with the default settings, the user proceeds to export the net to LaTeX (and to PDF, if selected). The export files are generated in the directory containing the Snoopy file for the net. Figure A.1 illustrates a typical application scenario.



(a) A Petri net in Snoopy



(b) Snoopy2LaTeX Export Dialog



(c) Generated Report

Figure A.1: Graphical illustration of a typical application scenario of Snoopy2LaTeX Export. The user creates or opens a snoopy net file (a), customizes the export properties in Snoopy2LaTeX dialog (b), and exports it to generate a human-readable report for the net (c).

Table A.1: Minimal and Optimal Hardware Requirements

Requirements	Minimal	Optimal
Processor	1 GHz	2 GHz
RAM	256 MB	1 GB
Free Hard Disk Space	500 MB	2 GB
LAN Adapter	X	X

A.3 System Requirements

In this section we outline the hardware and software requirements to run Snoopy2LaTeX on your computer.

Hardware Requirements

The hardware requirements depend on your specific needs. For instance, if you plan to run big models (100,000 to 1000,000 variables), then higher requirements are needed than to run relatively small ones. At the time of writing this manual, a 64Bit machine, preferably running native Linux is recommended for computational expensive models. Table A.1 outlines the minimum and the optimal hardware required to run Snoopy. Similar requirements apply for the use of its Snoopy2LaTeX Export.

Software Requirements

Snoopy2LaTeX Export is are platform-independent. Therefore you can use it on your favourite operating system. In more specific terms, you will need one of the following operating systems running on your computer:

- Windows XP or higher
- Mac OS X 10.5 or higher
- Linux (e.g. Ubuntu, Cent OS)

Next, you will need a copy of Snoopy suitable for your OS which can be downloaded from `http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy`.

If you wish to generate PDF report using Snoopy2LaTeX, you also need a TeX distribution to compile the generated LaTeX files. There are many TeX distributions available for different operating systems such as:

- **MiKTeX** for Windows
- **TeX Live** for Linux and other UNIX-like systems
- **MacTeX** redistribution of TeX Live for Mac OS X

A.4 Getting Started

A.4.1 Launching Snoopy2LaTeX

The first to use Snoopy2LaTeX Export is to open Snoopy on your machine. The home window of Snoopy is shown in Figure A.2. The exact procedure depends on the operating system you use. We assume that you have basic knowledge to run software on your operating system.

Figure A.2: Snoopy Home Window.

Next, you need to create the biochemical reaction network (that you wish to Export) using Snoopy, or open the corresponding Snoopy file, if it already exists.

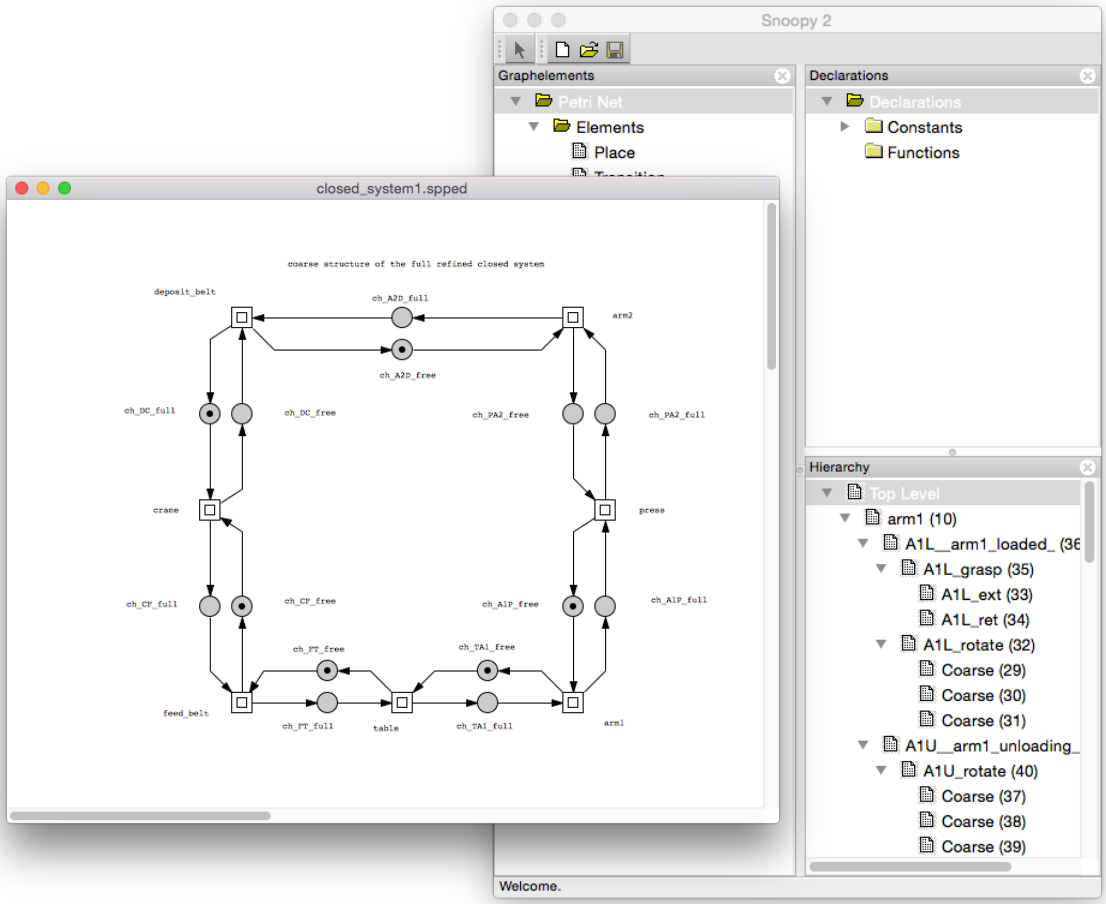


Figure A.3: Net file open in Snoopy

Once you have the net that you need to Export to LaTeX open in Snoopy, as illustrated in Figure A.3, follow these steps:

1. Go to **File** menu in the menu bar

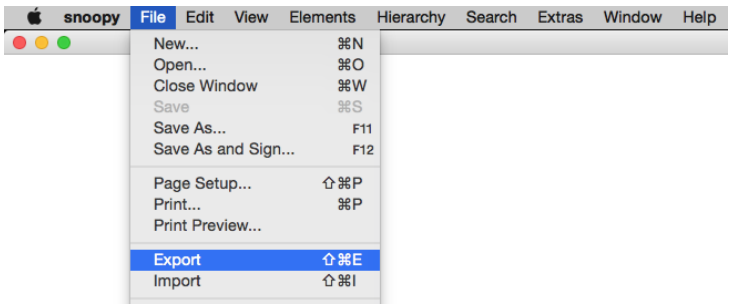


Figure A.4: Snoopy File Menu

2. Select **Export** from the drop down menu (as in Figure A.4)

3. A dialog listing all the exports allowed in Snoopy opens. From the checklist, select **Export to LaTex** (as in Figure A.5)

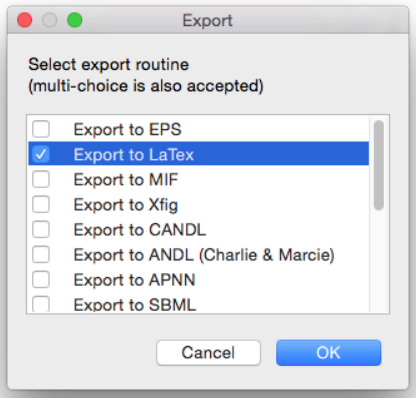


Figure A.5: Snoopy Export Dialog

The Snoop2LaTeXdialog is launched (as in Figure A.6).

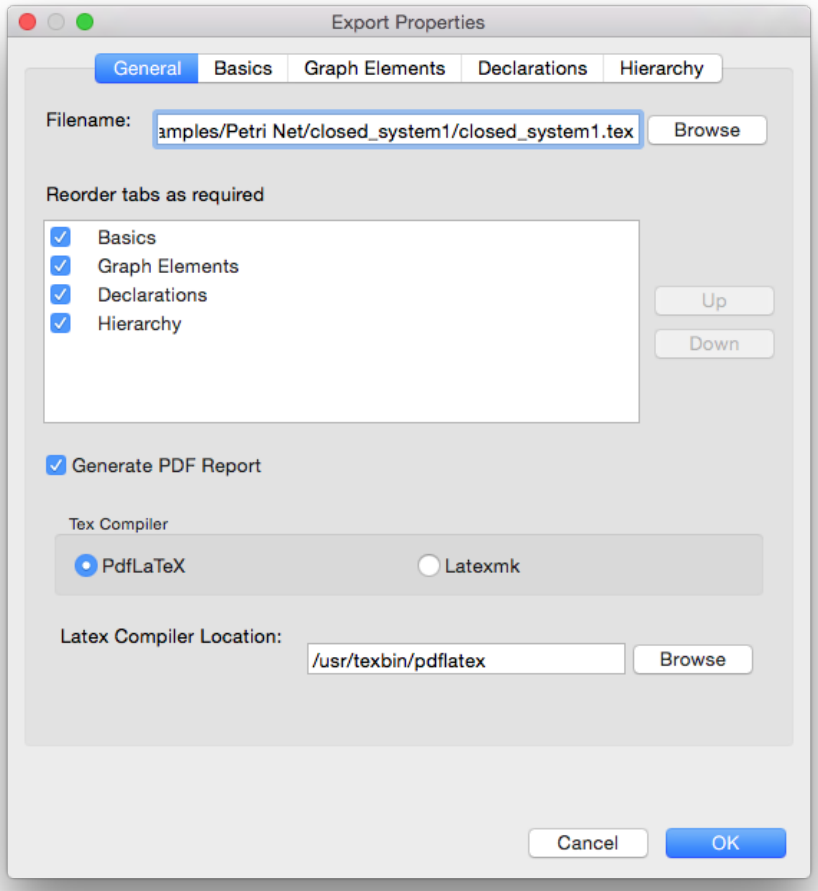


Figure A.6: Snoopy2LaTeX Export Dialog

A.4.2 Report Customization

Snoopy2LaTeX Export dialog provides several options for an exhaustive customization of the LaTeX report, as desired. Thus, it facilitates users to have a complete control over the content and layout of the generated report.

The Export dialog is organized as a tabbed interface (as shown in Figure A.6), with each tab corresponding to a specific domain of report. The following sub-sections render a detailed description of the various custom options offered by each tab.

A.4.2.1 General

This is the first tab in the dialog (as illustrated in Figure A.7). It corresponds to the general options for the export, which are unrelated to the properties of the given net.

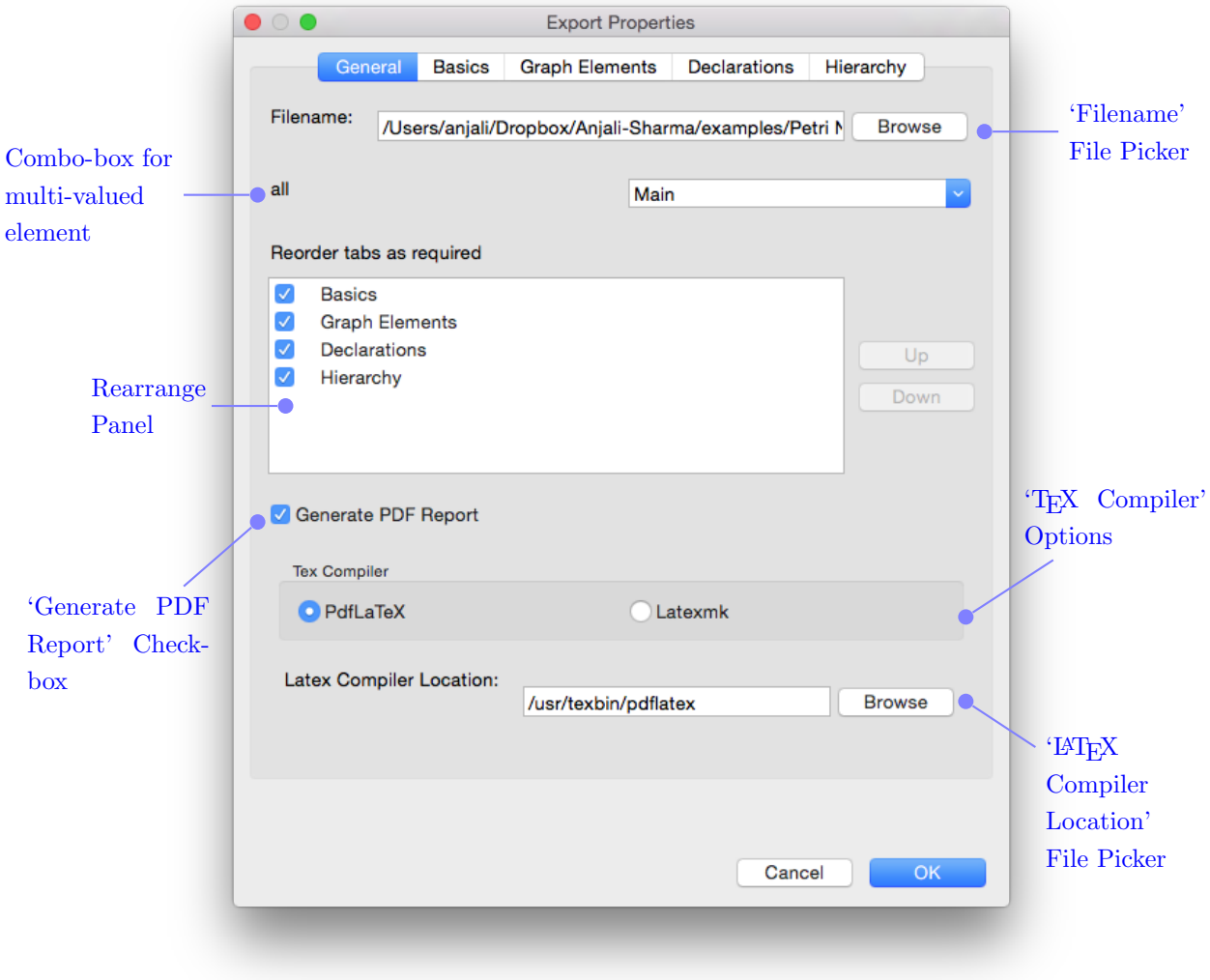


Figure A.7: Snoopy2LaTeX: General Tab

The utility of each element of General tab is described below:

- **'Filename' File Picker**

  The first element on the top is a file picker with a text field labeled **Filename** and a **Browse** button. The text field initially shows the default path for the output TeX file which is same as the directory of the corresponding Snoopy file. The path and filename, can however, be changed by manual entry into the text field or by using the browse option.

- **Combo boxes for multi-valued elements**

  If the given net have certain elements with mutiple value sets, then a combo box for each such element listing all its value sets appears below the filename field in the Export dialog (this is a general feature of Export dialog, for export to any format). Custom selections in these combo-boxes however *do not have any effect* for our Snoopy2LaTeXExport, as this export explicitly handles such elements. Examples of such multi-valued attributes are illustrated in Figure A.8.

- **Rearrange Panel**

  This panel lists the other four tabs (which correspond to sections of the report), i.e., *Basics, Graph Elements, Declarations* and *Hierarchy*. The ordering of the respective sections can be changed using the 'Up' and 'Down' buttons provided to the right. Further, the entire sections can be removed from the report by simply unchecking them from this list. By default, all of them are included into the generated report.

- **'Generate PDF Report' Checkbox**

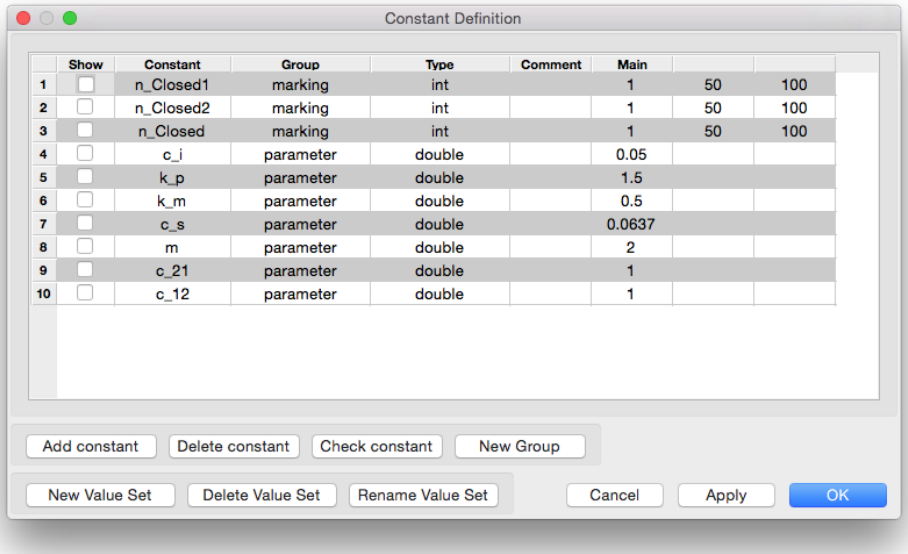
  This checkbox corresponds to the select/deselect option for the generation of PDF report. If the user just wishes to export the net to LaTeX code, the checkbox should be unchecked, else the generated LaTeX files are further compiled to generate resulting PDF, when exported.

- **'TeX Compiler' options**

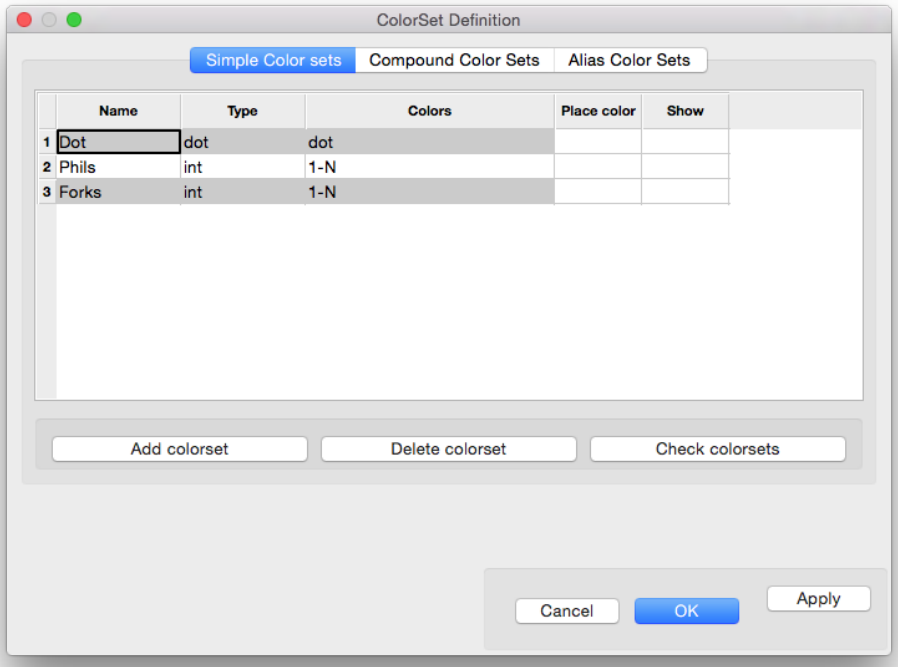
  If 'Generate PDF Report' option is selected, the user can chose between two basic TeX compiler chains (*pdflatex* and *latexmk*) to be used for the generation of PDF report.

- **'LaTeX Compiler Location' File Picker**

  If 'Generate PDF Report' option is selected, the user needs to provide the path of the LaTeX compiler (as selected in the radio button group above) in the system. The path can be given via manual entry into the text field or by browsing using the **Browse** button. By default, it contains the default installation path for the compiler.



(a) Constant Definition Dialog in Snoopy



(b) ColorSet Definition Dialog in Snoopy

Figure A.8: Multi-valued Elements in Snoopy. Figure (a) shows *Constants* for a sample Petri net. For colored Petri nets, besides other declaration elements, *Color Sets* are also multi-valued elements as depicted in (b).

A.4.2.2 Basics

This is the second tab in the dialog (as illustrated in Figure A.9). It corresponds to the basic attributes of the net and the output report, which are generic to any net class.

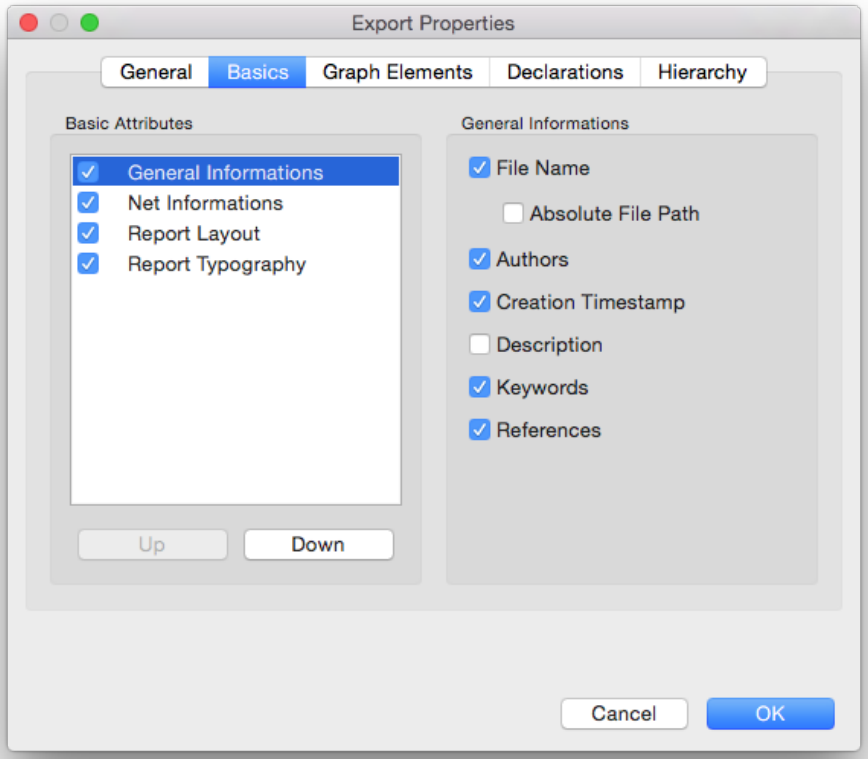


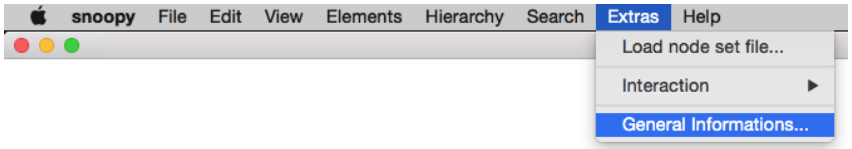
Figure A.9: Snoopy2LaTeX: Basics Tab

The Basics tab is organized into two panels. The *left panel* lists the basic elements for customization while the *right panel* shows the attributes corresponding to the selected element on the left panel. Selection/deselection of any element in the left panel enables/disables the corresponding right panel for customization respectively.

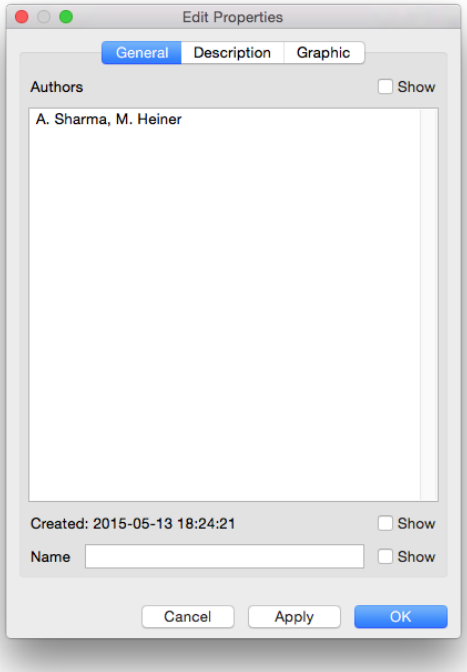
Further, all elements listed in the left panel can be reordered, as desired in the report, using the 'Up' and 'Down' buttons provided. The custom attributes for each element are described in the following paragraphs.

- **General Informations**

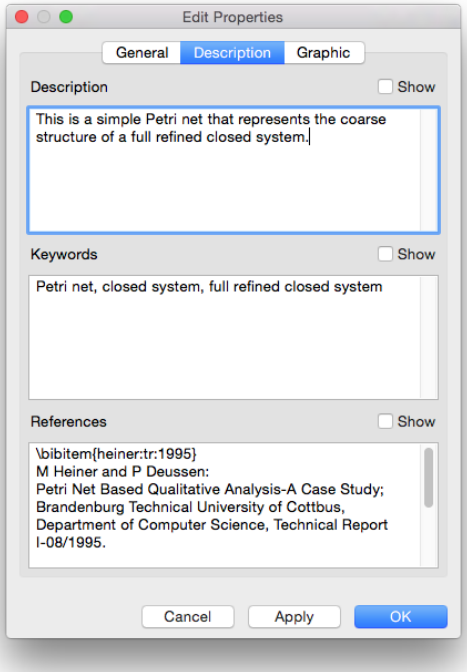
  This element of Snoopy2LaTeX Export corresponds to the 'General Informations' menu option of Snoopy (as illustrated in Figure A.10).



(a) Snoopy Menu: General Informations



(b) General Tab



(c) Description Tab

Figure A.10: Snoopy: **General Informations**. Figure (a) shows the General Informations option in the Snoopy menu. Figure(b) shows the General tab of the respective dialog with attributes (*Authors, Creation Timestamp* and *Name*) for customization. Figure (c) shows the Description tab of the dialog with attributes (*Description, Keywords* and *References*) for customization.

Described below are the corresponding attributes under General Informations in Export Dialog, as illustrated in Figure A.9. Each attribute can be checked/unchecked to include/exclude them into/from the generated report respectively.

1. **Filename**

   It exports the name of the net, which is identified by a user-defined name and a file extension corresponding to the net class (e.g. closed_system.spped).

2. **Absolute File Path**

   It exports the complete path of the net file in the user system (e.g. /Users/anjali/Petri Net/closed_system.spped).

3. **Authors**

   It exports the name(s) of author(s) of the net, as specified by the user in the 'Authors' field of General Informations dialog.

4. **Creation Timestamp**

   It exports the timestamp (*date in YYYY-MM-DD format and time in HH:MM:SS format*) corresponding to the creation of the net, as displayed beside the 'Created' label in the General Informations dialog.

5. **Description**

   It exports the description for the net, as specified by the user in the 'Description' field of General Informations dialog.

6. **Keywords**

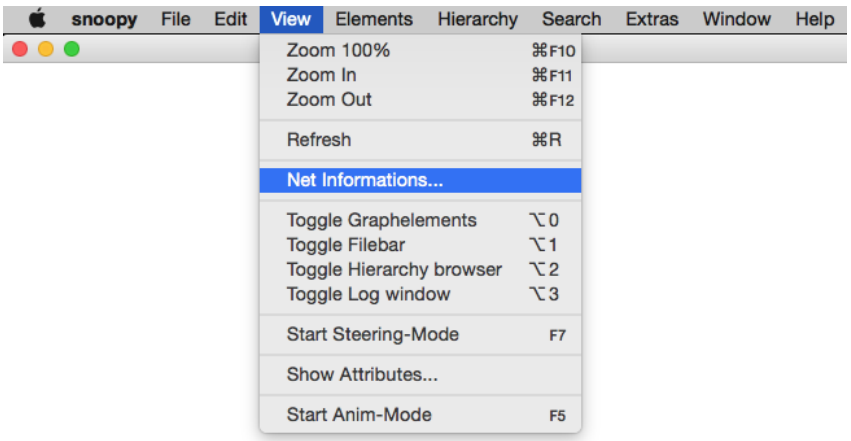
   It exports the keywords for the net, as specified by the user in the 'Keywords' field of General Informations dialog.
   [NOTE: *It is assumed that keywords are entered as comma-separated strings, as shown in the 'Keywords' field of Figure A.10(c).*]

7. **References**

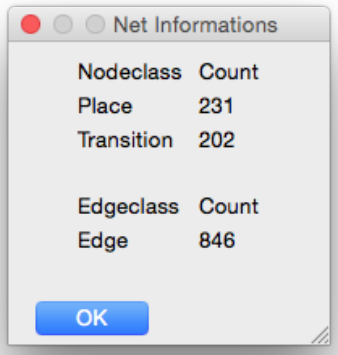
   It exports the references for the net, as specified by the user in the 'References' field of General Informations dialog.
   [NOTE: *It is assumed that references are entered bibitems [3], as shown in the 'References' field of Figure A.10(c).*]

- **Net Informations**

  This element of Snoopy2LaTeX Export corresponds to the 'Net Informations' menu option of Snoopy (as illustrated in Figure A.11).



(a) Snoopy Menu: Net Informations



(b) General Tab

Figure A.11: Snoopy: **Net Informations**. Figure (a) shows the Net Informations option in the Snoopy menu. Figure(b) shows the corresponding dialog with the element count for all elements of the given net.

Described below are the corresponding attributes under Net Informations in Export Dialog, as illustrated in Figure A.12. Each attribute can be checked/unchecked to include/exclude them into/from the generated report respectively.

1. **Net Class**

   It exports all net classes (all *Node classes* and *Edge classes*) for the given net.

2. **Element Count**

   It exports the corresponding element count for each net class for the given net.

- **Report Layout**

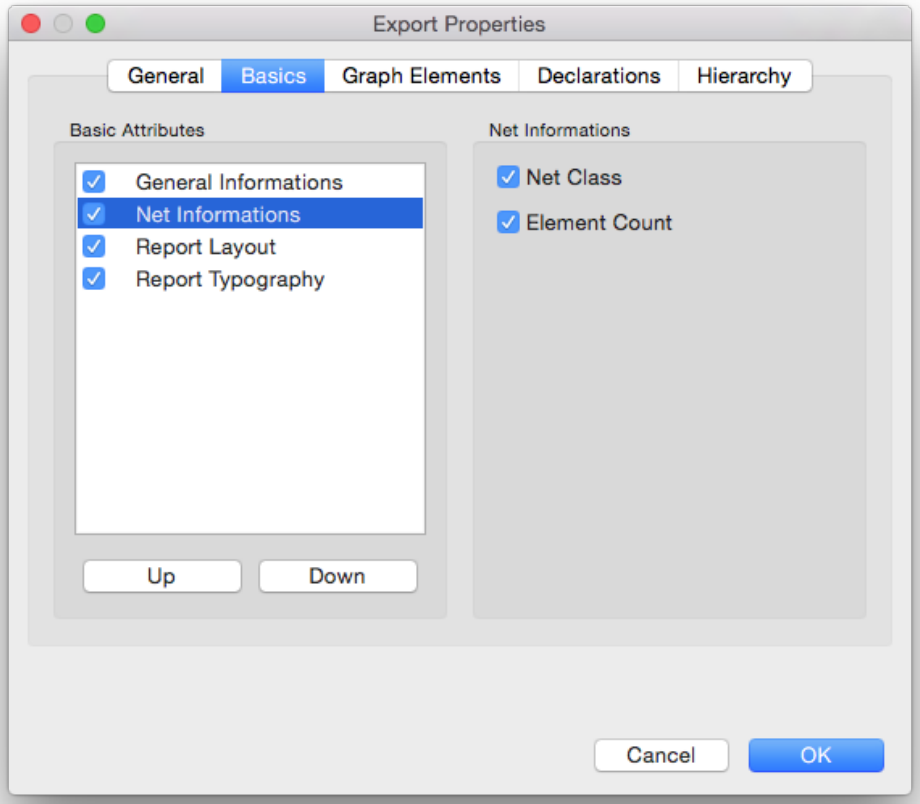


Figure A.12: Snoopy2LaTeX: Net Informations

This element of Snoopy2LaTeX Export corresponds to a set of custom attributes that define the layout of the generated report.

Described below are the layout attributes under Report Layout in Export Dialog, as illustrated in Figure A.13. Except orientation, each attribute can be checked/unchecked to include/exclude them into/from the generated report respectively.

1. **Orientation**
   It defines the page orientation for the generated report. It can be switched between *portrait* and *landscape* as desired by the user. By default, it is *portrait*.
2. **Insert Page Numbers**
   It adds page numbering to the complete report. The page numbers are inserted as a right-aligned footer.
   (*Format: Page <Current Page> of <Total Page Count>*)
3. **Insert Date**
   It inserts current date (system date when the net is exported) as a left-aligned footer to the complete report.
   (*Format: DD/MM/YYYY*)

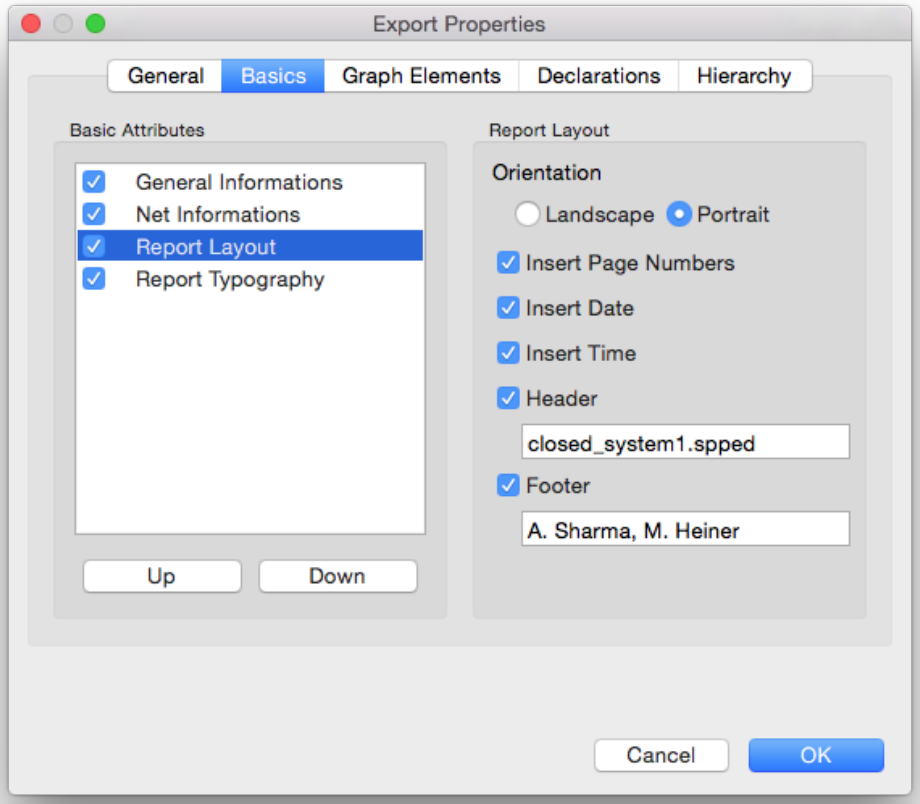


Figure A.13: Snoopy2LaTeX: Report Layout

4. **Insert Time**
   It inserts current time (system time when the net is exported) as a left-aligned footer to the complete report.
   (*Format: HH:MM*)
5. **Header**
   It includes a text field for the user to specify a header for the report. By default, the filename is set as the header.
6. **Footer**
   It includes a text field for the user to specify a footer for the report. By default, the author(s) is/are set as the footer.

- **Report Typography**

  This element of Snoopy2LaTeX Export corresponds to a set of custom attributes that define the typography of the generated report.

  Described below are the typography attributes under Report Typography in Export Dialog, as illustrated in Figure A.14. Each attribute has a range of options as an expandable list to chose from.

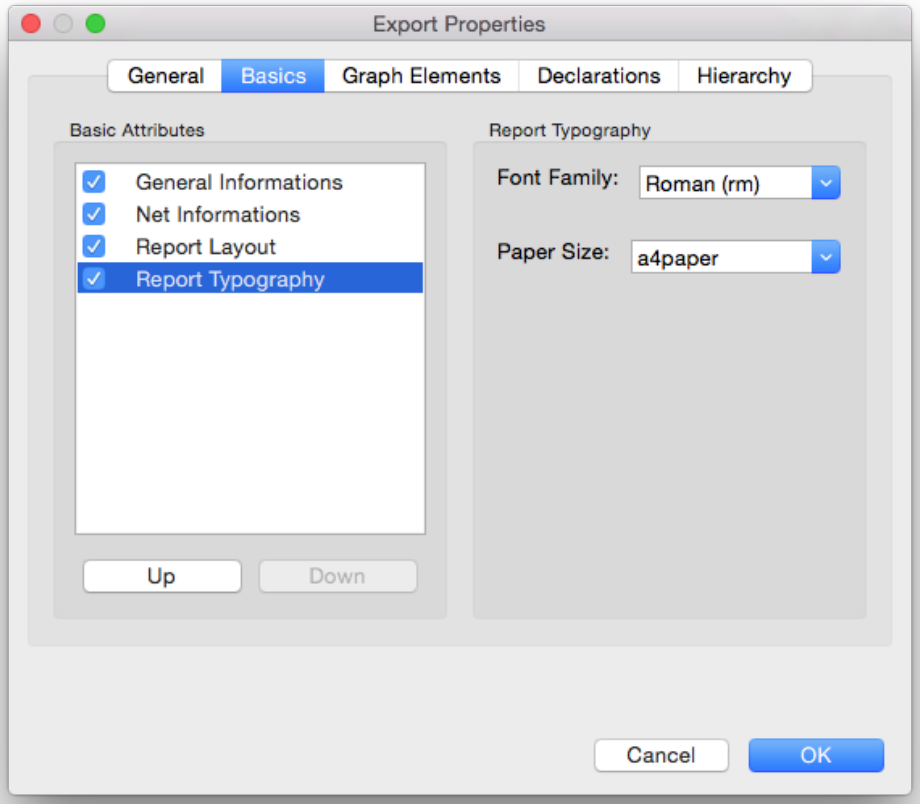


Figure A.14: Snoopy2LaTeX: Report Typography

1. **Font Family**
   It allows the user to chose the font family (*roman, sans serif* or *monospace*) to be used for the LaTeX report, from the list. By default, *roman* is used.
2. **Paper Size**
   It allows the user to specify the paper size for the LaTeX report. The user can chose from the range of pre-defined page sizes (specific to LaTeX) listed. By default, the paper size is standard *a4 paper*.

A.4.2.3 Graph Elements

This is the third tab in the dialog (as illustrated in Figure A.15). It corresponds to the custom attributes that are specific to the graph elements for the given net.

The Graph Elements tab, like the Basics tab is organized into two panels. The *left panel* lists the graph elements for the given net, while the *right panel* shows the attributes and nodes corresponding to the selected element on the left panel. Selection/deselection of any element in the left panel enables/disables the corresponding right panel for customization respectively.

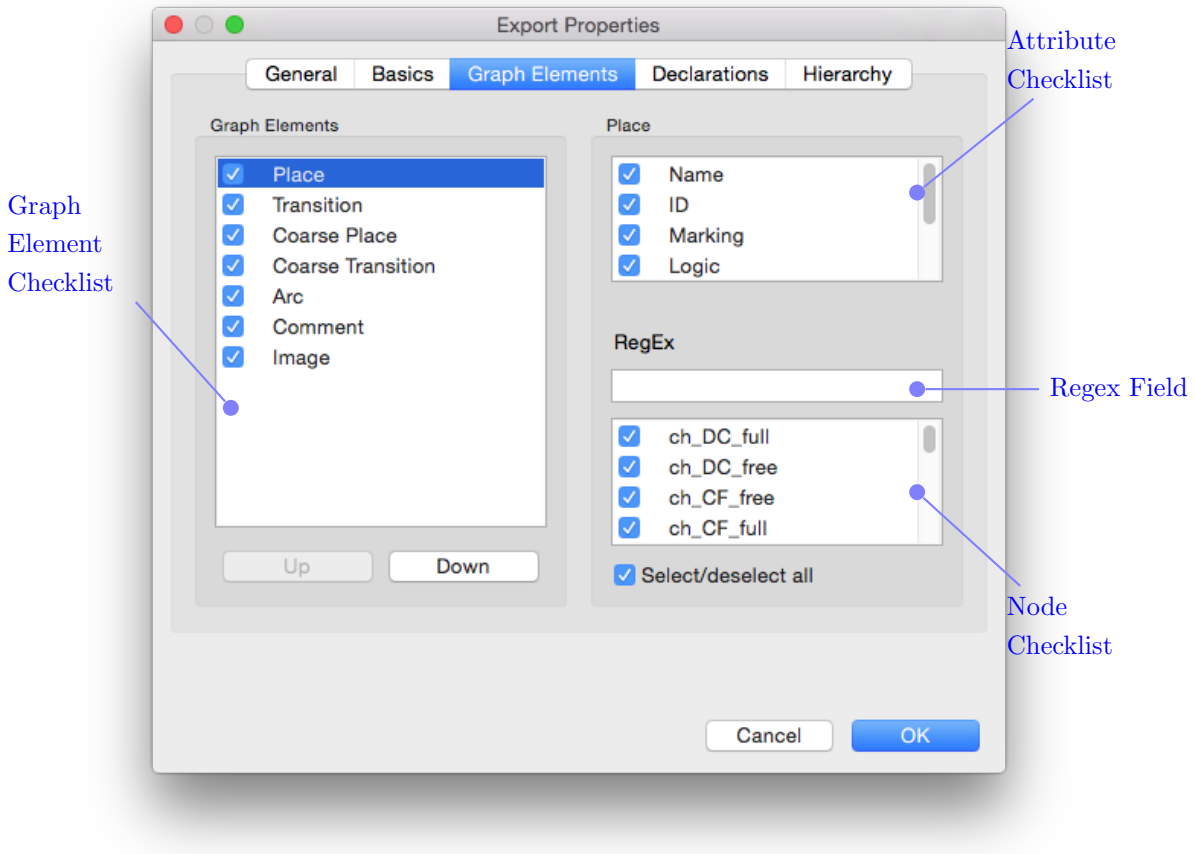


Figure A.15: Snoopy2LaTeX: Graph Elements Tab. The labels indicate the four generic parts of the tab. The current view lists options for a simple Petri net example.

However, unlike the Basics tab, the contents of this tab *vary with the net class for the given net*. The generic structure of this tab can be categorized into four main parts:

1. **Graph Element Checklist**

   Each net class has a unique set of graph elements. The left panel contains a checklist of graph elements for the given net. For example, in Figure A.15, the graph elements for a simple Petri net class are listed.

   By default, all elements are selected for export. The user can deselect any graph element from the list, to avoid exporting it. Also, these elements can be reordered, as desired in the report, using the 'Up' and 'Down' buttons below the list.

2. **Attribute Checklist**

   Every graph element has a unique set of attributes. This top right panel lists the attributes for the current selection of graph element. For example, in Figure A.15, the attributes for 'Place' element for are listed.

   Every element has a default set of attributes selected for export. The user can select/deselect any attribute from the list, to include/exclude it into/from the generated report respectively.

3. **Node Checklist**

   This bottom right panel lists the nodes belonging to the current selection of graph element. For example, in Figure A.15, the nodes of 'Place' element type for the given net are listed.

   By default, all nodes are selected for export. The user can select/deselect any node from the list, to include/exclude it into/from the report respectively. Besides, the complete node list can be selected/deselected by a single click using the **Select/deselect all** checkbox.

4. **Regex Field**

   The **RegEx** labeled text field on the right panel allows the user to enter a regex (or regular expression). As soon as a valid regex is entered, the node names in the 'Node Checklist' corresponding to this expression get selected, deselecting others. *To learn more about regex, please refer section A.6.*

For **Arcs**, the layout of this tab varies a bit (shown in Figure A.16). Instead of the Regex field and arc checklist (because having a checklist for arcs makes little sense), the tab has:

- **Order By** option with radio buttons for *Source* and *Target*. This allows the user to order the arcs in report by their source node name or target node name, respectively. By default, they are *ordered by source*.
- **Group By** option with radio buttons for *Place to Transition* and *Transition to Place*. If the user selects 'Place to Transition' grouping, then all the arcs directed from places to transitions are ordered and listed first in the table; remaining arcs (belonging to the Transition to Place group) follow. Similarly, if the other option is selected, arcs for this group precede the other arcs in the list. By default, arcs are *grouped by place to transition*.

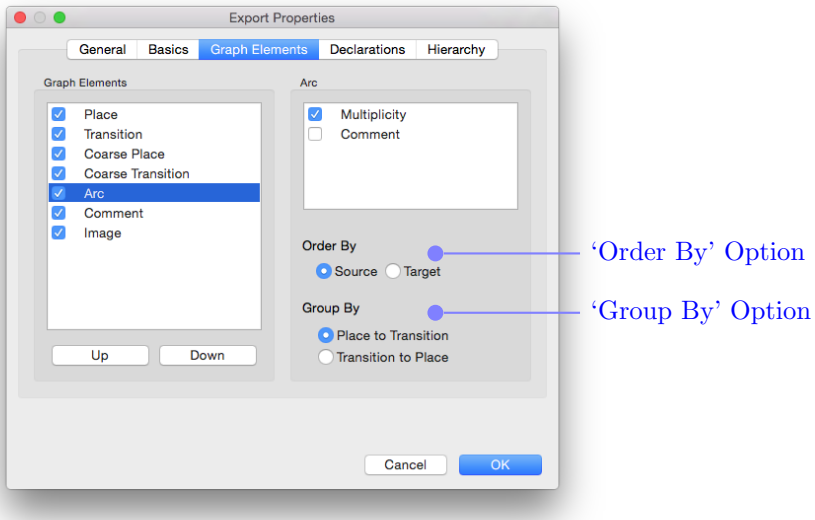


Figure A.16: Snoopy2LaTeX: Graph Elements tab for Arcs

A.4.2.4 Declarations

This is the fourth tab in the dialog (as illustrated in Figure A.17). It corresponds to the custom attributes that are specific to the graph elements for the given net.

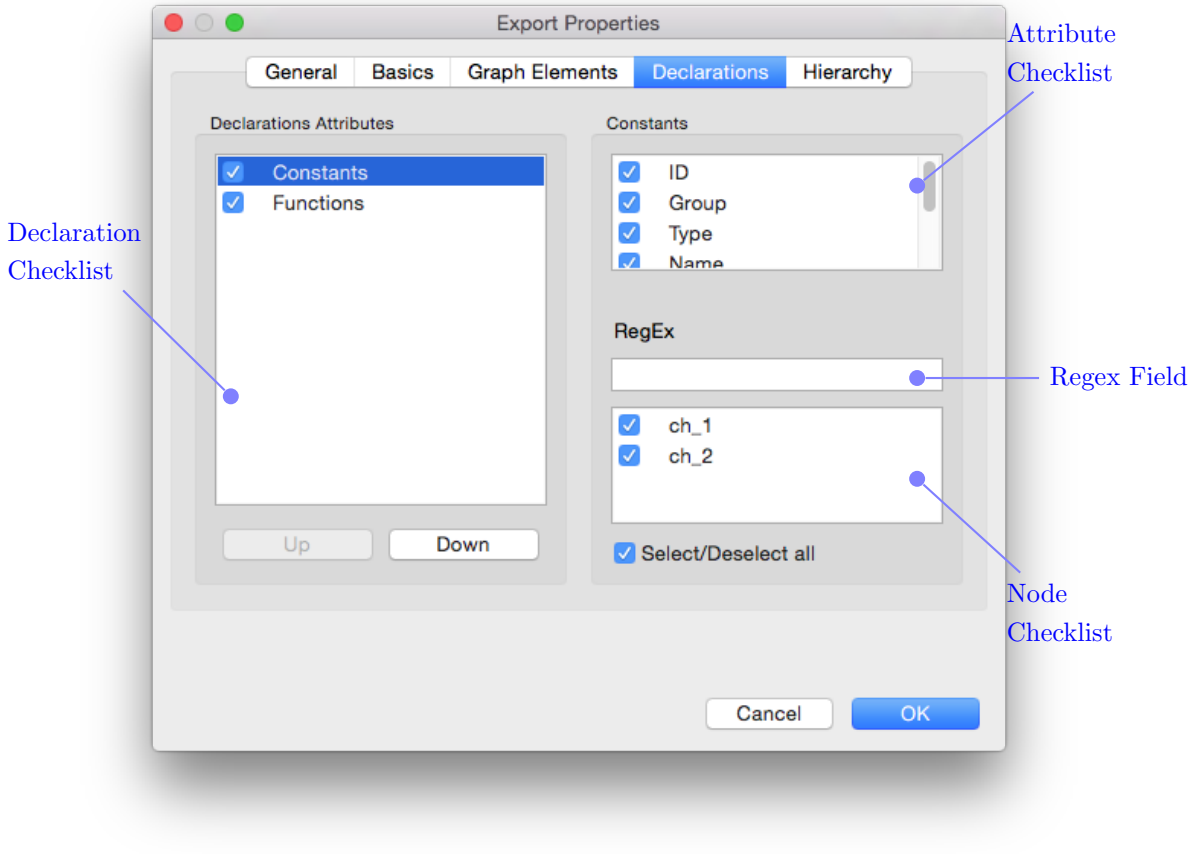


Figure A.17: Snoopy2LaTeX: Declarations Tab. The labels indicate the four generic parts of the tab. The current view lists options for a simple Petri net example.

The Declarations tab also, like the Basics tab, is organized into two panels. The *left panel* lists the declarations for the given net, while the *right panel* shows the attributes and nodes corresponding to the selected element on the left panel. Selection/deselection of any element in the left panel enables/disables the corresponding right panel for customization respectively.

However, similar to the Graph Elements tab, the contents of this tab *vary with the net class for the given net*. The generic structure of this tab can be categorized into four main parts:

1. **Declarations Checklist**

   Each net class has a unique set of declarations. The left panel contains a checklist of declarations for the given net. For example, in Figure A.17, the declarations for a simple Petri net class are listed.

   By default, all declarations are selected for export. The user can deselect any declaration element from the list, to avoid exporting it. Also, these elements can

be reordered, as desired in the report, using the 'Up' and 'Down' buttons below the list.

2. **Attribute Checklist**

   Every declaration element has a unique set of attributes. This top right panel lists the attributes for the current selection of declaration. For example, in Figure A.17, the attributes for 'Constants' are listed.

   Every element has a default set of attributes selected for export. The user can select/deselect any attribute from the list, to include/exclude it into/from the generated report respectively.

3. **Node Checklist**

   Every net has a unique set of nodes corresponding to each declaration. This bottom right panel lists the nodes for the current selection of declaration. For example, in Figure A.17, the nodes of 'Constants' declaration type for the given net are listed.

   By default, all nodes are selected for export. The user can select/deselect any node from the list, to include/exclude it into/from the generated report respectively. Besides, the complete node list can be selected/deselected by a single click using the **Select/deselect all** checkbox.

4. **Regex Field**

   This **RegEx** field works similar to the regex field of *Graph Elements* tab, for its respective node list.

A.4.2.5 Hierarchy

This is the last tab in the dialog (as illustrated in Figure A.18). It corresponds to the export properties for the hierarchical structure of the given net.

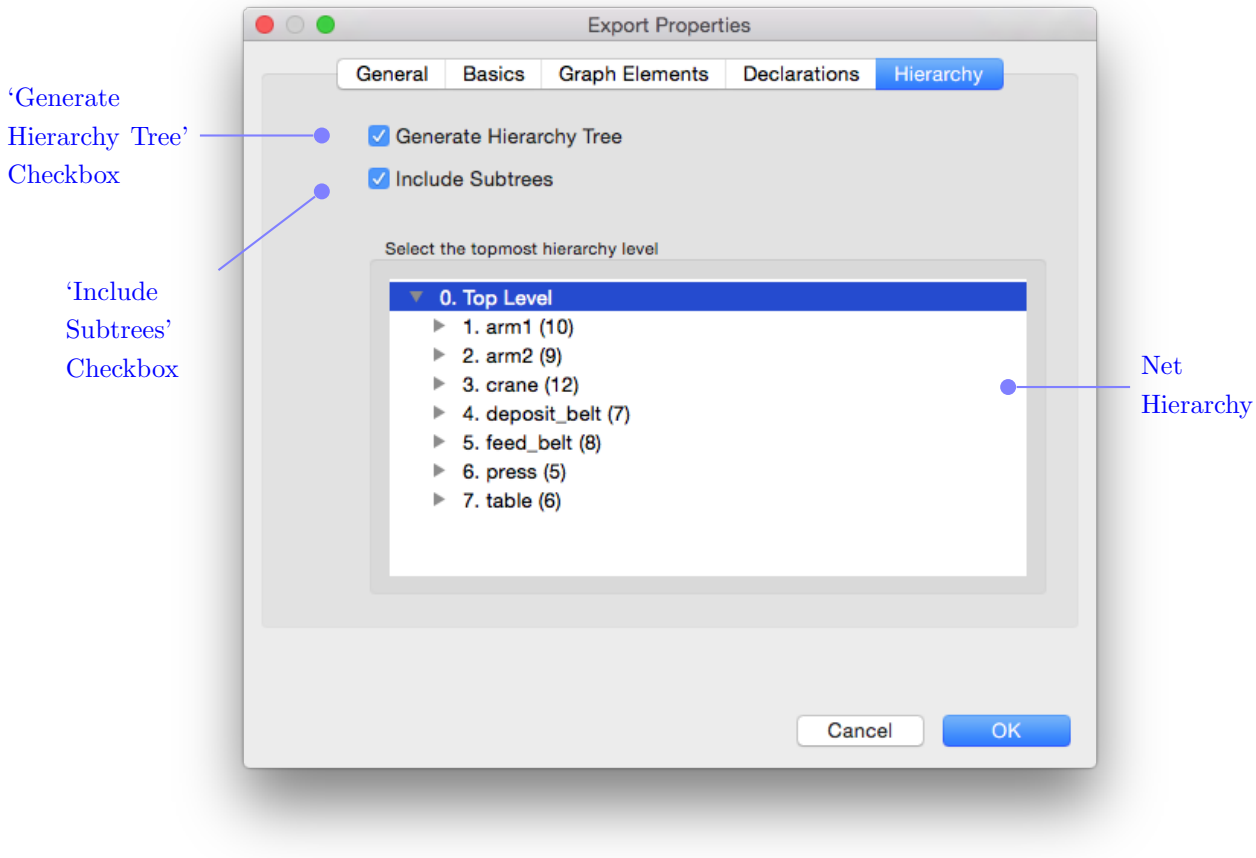


Figure A.18: Snoopy2LaTeX: Hierarchy Tab. The label indicates the generic panel for net hierarchy. The current view shows the net hierarchy for a simple Petri net example.

The generic structure for this tab can be categorized into three main parts:

1. **Net Hierarchy**

   This panel shows the hierarchical structure for the given net. The user can select single or multiple levels from this hierarchy and the nets corresponding to these levels shall be exported to the generated report. To export all subnets in the net hierarchy, select **Top Level**.

   [NOTE: *If a level and one or more of its sublevels are selected for export, only the common ancestor to all such levels is considered for the final export.*]

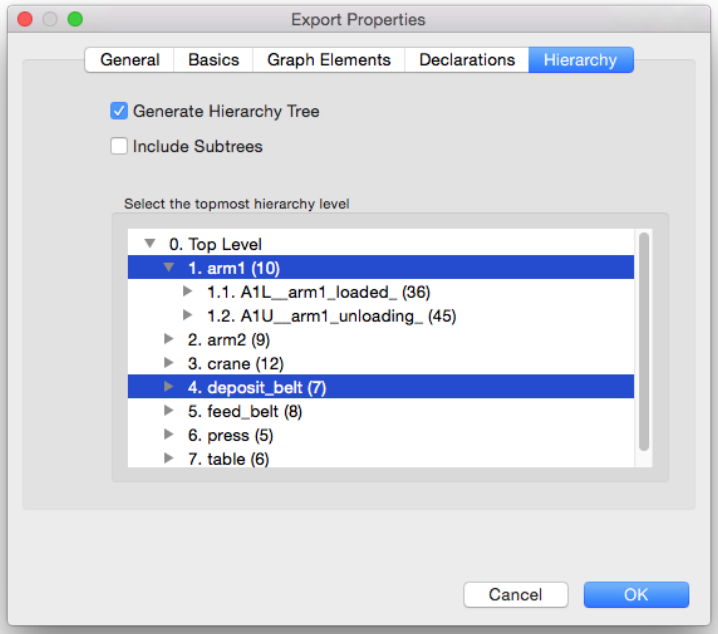
2. **'Generate Hierarchy Tree' Checkbox**

   This option, if selected, shall render a tree like structure for each selection of hierarchy level, thus graphically illustrating the hierarchical positioning of the corresponding net and its subnets (if 'Include Subtrees' selected). Figure A.19 and A.20 illustrate this feature.

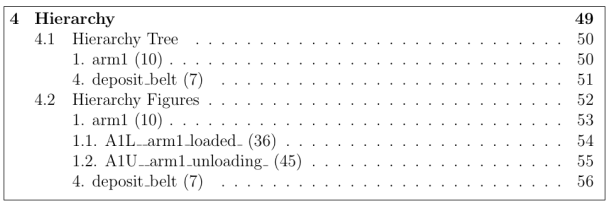
3. **'Include Subtrees' Checkbox**

   If this option is selected, then corresponding to each selection of hierarchy level, all its sub-levels (whether expanded/collapsed in the 'Net Hierarchy' panel) are also exported to the report (and are included in the hierarchy tree for the level, if 'Generate Hierarchy Tree' option is selected). Figure A.20 illustrates this feature.

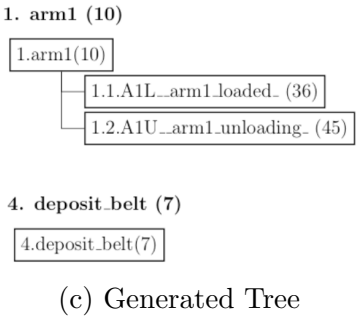
   If this option is unchecked, only the immediate selections for hierarchy levels and the corresponding sub-levels expanded in the 'Net Hierarchy' panel are exported. *Therefore, if this option is not selected, then for a selected level, the user must explicitly expand the sub-levels he needs to export.* Figure A.19 illustrates this feature.



(a) Hierarchy Tab

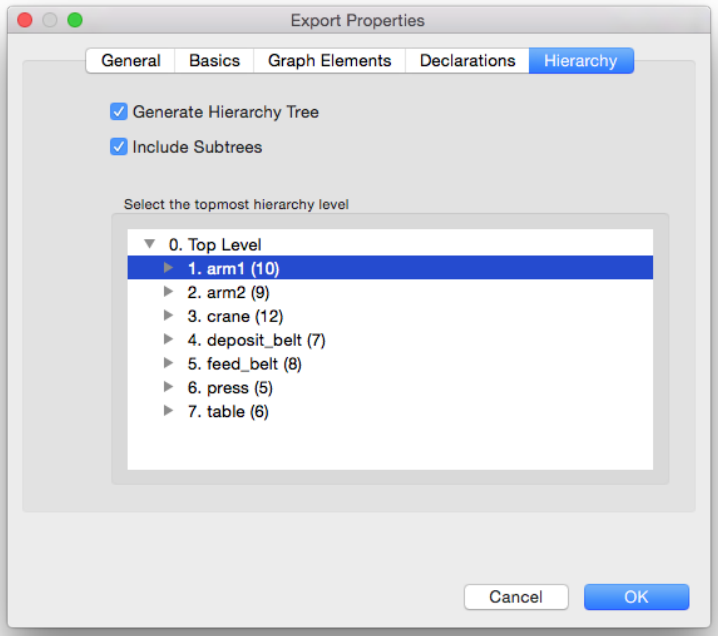


(b) Exported Hierarchy Structure



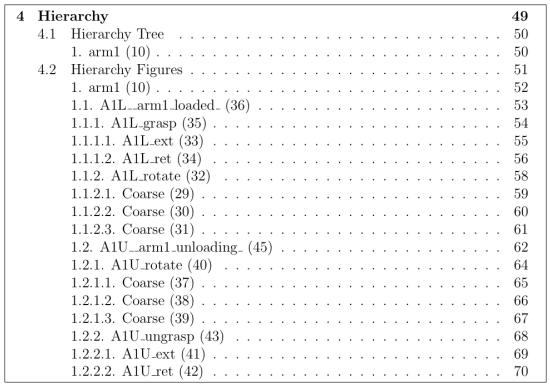
(c) Generated Tree

Figure A.19: Hierarchy Export: 'Include Subtrees' Deselected. If the 'Include Subtrees' option is deselected (a), then the generated report exports only the selected levels and their expanded sublevels, as illustrated by the table of contents for the resulting report (b).



(a) Hierarchy Tab

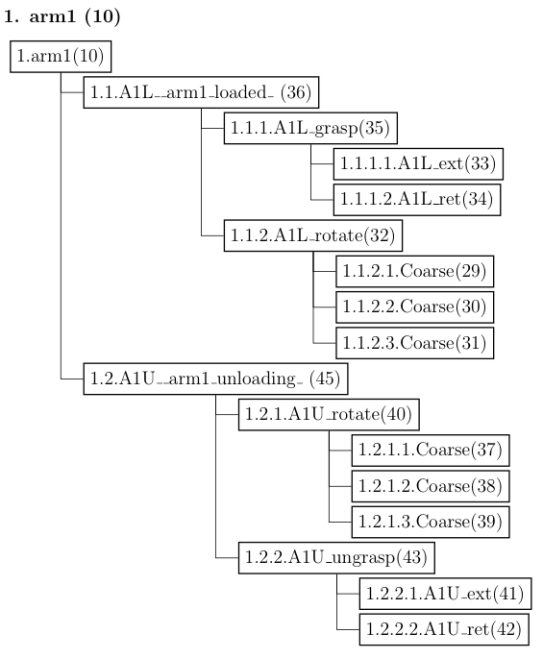
↓



4 Hierarchy **49**

4.1 Hierarchy Tree . . . 50
1. arm1 (10) . . . 50
4.2 Hierarchy Figures . . . 51
1. arm1 (10) . . . 52
1.1. A1L_arm1_loaded_ (36) . . . 53
1.1.1. A1L_grasp (35) . . . 54
1.1.1.1. A1L_ext (33) . . . 55
1.1.1.2. A1L_ret (34) . . . 56
1.1.2. A1L_rotate (32) . . . 58
1.1.2.1. Coarse (29) . . . 59
1.1.2.2. Coarse (30) . . . 60
1.1.2.3. Coarse (31) . . . 61
1.2. A1U_arm1_unloading_ (45) . . . 62
1.2.1. A1U_rotate (40) . . . 64
1.2.1.1. Coarse (37) . . . 65
1.2.1.2. Coarse (38) . . . 66
1.2.1.3. Coarse (39) . . . 67
1.2.2. A1U_ungrasp (43) . . . 68
1.2.2.1. A1U_ext (41) . . . 69
1.2.2.2. A1U_ret (42) . . . 70

(b) Exported Hierarchy Structure



(c) Generated Tree

Figure A.20: Hierarchy Export: 'Include Subtrees' Selected. If the 'Include Subtrees' option is selected (a), then the generated report exports the selected levels as well as all its sub-levels, as illustrated by the table of contents for the resulting report (b).

A.4.3 Export

Once all the attributes for the given net are customized by the user as desired, the next step is to export the net. Snoopy2LaTeX offers a two-step export functionality with the option to:

- **Export to LaTeX** code, that can be further compiled using a TeX compiler to generate a human-readable report (such as PDF or DVI) suitable for printing or digital distribution.
- **Export to PDF** by implicit compilation of generated LaTeX files.

The following sections explain the steps to retrieve the generated report files.

A.4.3.1 Export to LaTeX

To export the net to LaTeX, the user must follow these steps:

1. Specify the main LaTeX file name and path in the **Filename** field of *General* tab.
2. To export, press 'OK' button in the dialog.
3. To cancel the export, press 'Cancel' button in the dialog.

Once the OK button is pressed, the net with the custom settings is exported to a collection of interlinked LaTeX files. *For a detailed description of the generated report structure, please refer section A.5.*

A.4.3.2 Export to PDF

To export the net to PDF, the user must follow these steps:

1. Check the 'Generate PDF Report' checkbox in the *General* tab.
2. Choose between TeX compilers (*pdflatex* and *latexmk*) using radio buttons.
3. Specify the compiler path in user system, in the **Latex Compiler Location** field.
4. To export, press 'OK' button in the dialog.
5. To cancel the export to PDF, uncheck the 'Generate PDF Report' checkbox.
6. To cancel the complete export, press 'Cancel' button in the dialog.

Once the OK button is pressed, the net with the custom settings is exported to a collection of interlinked LaTeX files. These files are then compiled using the selected TeX compiler to generate a PDF report. *For a detailed description of the generated report structure, please refer section A.5.*

Please note that the generated LaTeX files contain an extensive cross-referencing among its elements, which increases with the increasing number of nodes or hierarchical complexity of the net. Besides, the resolution of cross-references primarily affects the compile time for a given LaTeX code. *Hence, for large nets, Export to PDF may take a significant amount of time to generated the PDF report.*

A.5 Generated Report Structure

This section shall render a detailed description of the generated report, which includes the generated LaTeX code and the corresponding PDF.

The generated LaTeX code is structured as a collection of interlinked LaTeX files. The main features of this structure are:

- **Standard report format**

  The generated report follows a standard format, with a structural flow as:

  1. Title Page
  2. Table of Contents
  3. Sections specific to net information
  4. References
  5. Glossary

  The template for LaTeX code generated can be given as:

```
\documentclass[pdftex,12pt]{article}
<preamble>

\begin{document}

\input{./TitlePage.tex}

\newpage
\tableofcontents

\input{./Basics.tex}
\input{./GraphElements.tex}
\input{./Declarations.tex}
\input{./Hierarchy.tex}
\input{./References.tex}
\input{./Glossary.tex}

\end{document}
```

  Even the default layout and typography settings for report follow the standard convention with *portrait* orientation, *roman* font family and *a4* paper size.

Contents

Figure A.21: Generated Report: Table of Contents. This illustrates the structure of generated report for a simple Petri net example. Each section corresponds to a tab of an export domain and each subsection corresponds to an element of that domain.

- **Modular structure**

  The generated LaTeX code is split across several small interlinked files, rather than a single large file. Each file serves as a self-contained collection of information pertaining to a specifc export element. This makes the code more **readable** and easy to **reuse**, **edit** and **debug**.

  As illustrated in the table of contents for a sample report in Figure A.21, the report is organized as a collection of sections and subsections for each export element. The standard report (with default custom options) has six sections:

  1. **Basics**

     This section contains all basic information for the given net, corresponding to the elements and attributes listed in the *Basics* tab of Export dialog. It primarily includes subsections *General Informations* and *Net Informations* (if selected for export).

     (a) **General Informations**
     This subsection contains the 'General Informations' for the given net as customized in the Export dialog. An example of this export is illustrated in Figure A.22.

     (b) **Net Informations**
     This subsection contains the 'Net Informations' for the given net as customized in the Export dialog. An example of this export is illustrated in Figure A.23.

  2. **Graph Elements**

     This section contains information about the graph elements for the net, corresponding to the elements, attributes and nodes listed in the *Graph Elements* tab. Each graph element type (selected for export) forms a subsection containing information for nodes of this type.

     Figures A.24 and A.25 show how the contents of this section vary for different net classes. Figure A.24 shows the table of contents for a simple Petri net, while Figure A.25 shows the table of contents of this section for a Stochastic Petri net.

  3. **Declarations**

     This section contains information about the declarations for the net, corresponding to the elements, attributes and nodes listed in the *Declarations* tab. Each declaration type (selected for export) forms a subsection containing information for nodes of this type, similar to the graph elements.

     Figures A.26 and A.27 show how the contents of this section vary for different net classes. Figure A.26 shows the table of contents for a simple Petri net, while Figure A.27 shows the table of contents of this section for a Colored Petri net.

4. **Hierarchy**

   This section contains information about the hierarchy of the net, corresponding to the hierarchy levels selected for export in the *Hierarchy* tab. This section has two subsections:

   (a) **Hierarchy Tree**
   This subsection corresponds to the 'Generate Hierarchy Tree' option of the Export dialog. Therefore, if this option is selected, this section consists of one hierarchy tree for each selected level. Figure A.28 shows an example of a hierarchy tree generated for a given level.

   (b) **Hierarchy Figures**
   This subsection consists of exported net figures for each selected level of net hierarchy. Figure A.29 shows an example of a net figure generated for a given level.

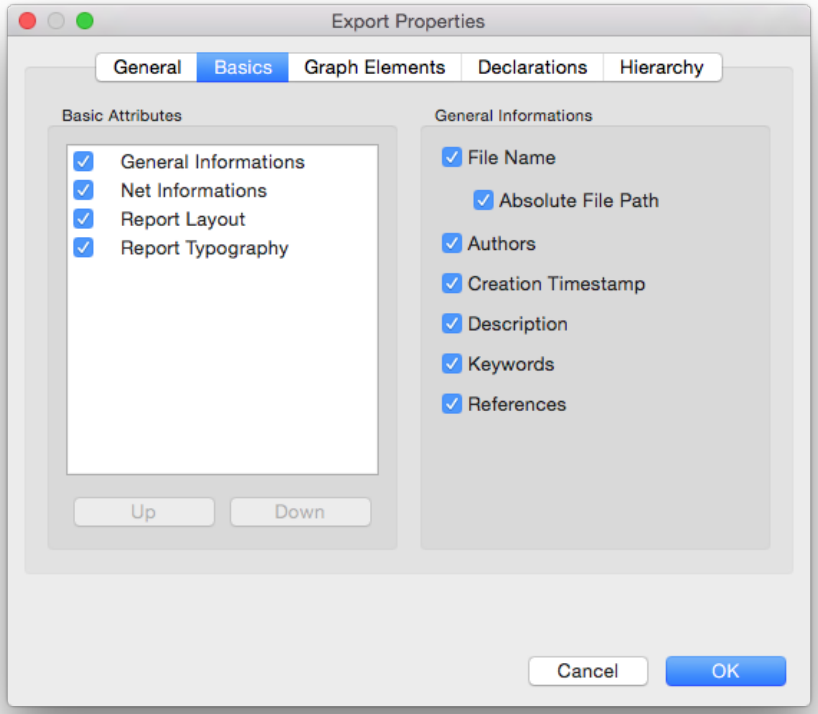
5. **References**

   This section contains the references for the given net, as included in the 'References' section of *General Informations* (illustrated in Figure A.30). Besides, every generated report contains two basics references [1] and [2].

6. **Glossary**

   This section contains a glossary of all the abbreviations used in the report. This is same of any net.

LaTeX code corresponding to each such section and subsection is placed in a separate file, named after the respective element name. This renders a code structure that is highly robust with respect to future use.



(a) Snoopy Menu: General Informations

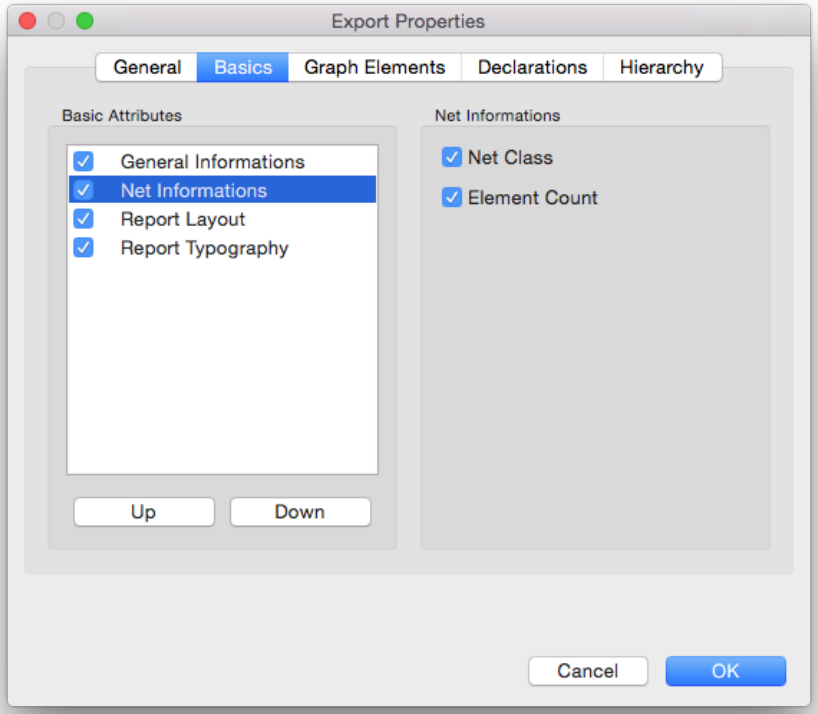
↓

1.1 General Informations

File Name: closed_system.spped
File Path: /Users/anjali/Dropbox/Anjali-Sharma/examples/PetriNet/closed_system/closed_system.spped
Authors: A. Sharma, M. Heiner
Creation Timestamp: 2015-01-22 15:24:03
Description: This is a simple Petri net that represents the coarse structure of a full refined closed system.
Keywords: Petri net, closed system, full refined closed system
References: [3, 4]

(b) Generated Report: General Informations

Figure A.22: Exported General Informations. All the selected attributes in the dialog (a) are exported to the report (b). The values for these attributes here correspond to the example net *closed_system.spped*, as shown in Figure A.10(b) and A.10(c).



(a) Snoopy Menu: Net Informations

↓

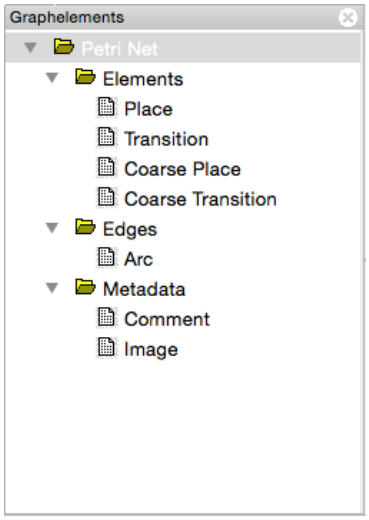
1.2 Net Informations

Net Class: Petri Net

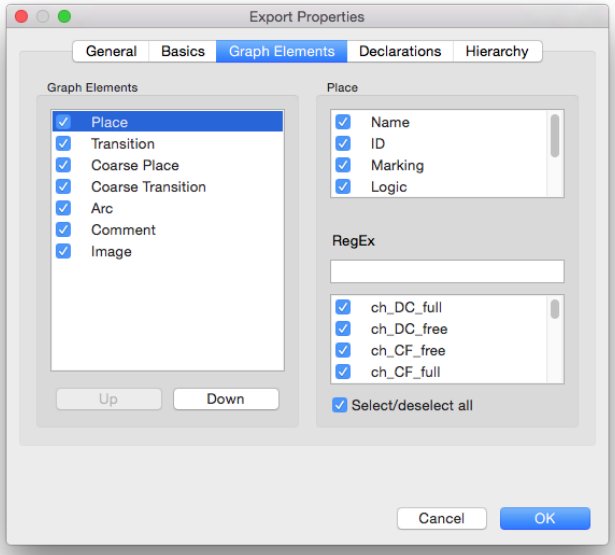
Element	Count
Place	231
Transition	202
Edge	846

(b) Generated Report: Net Informations

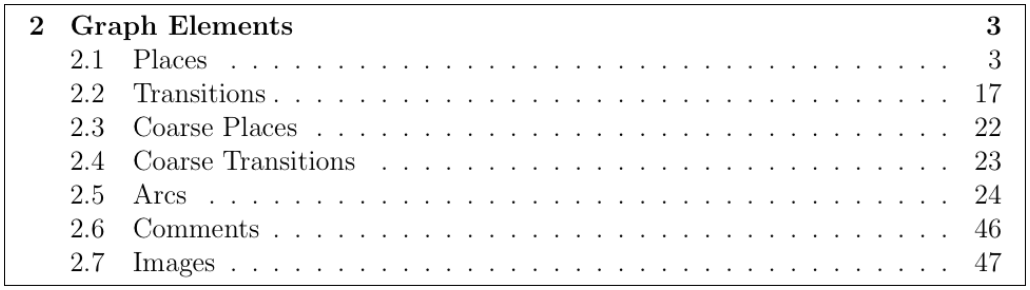
Figure A.23: Exported Net Informations. All the selected attributes in the dialog (a) are exported to the report (b). The values for these attributes here correspond to the example net *closed_system.spped*, as shown in Figure A.11(b).



(a) Snoopy: Graph Elements Panel



(b) Snoopy2LaTeX: Graph Elements for Petri Net

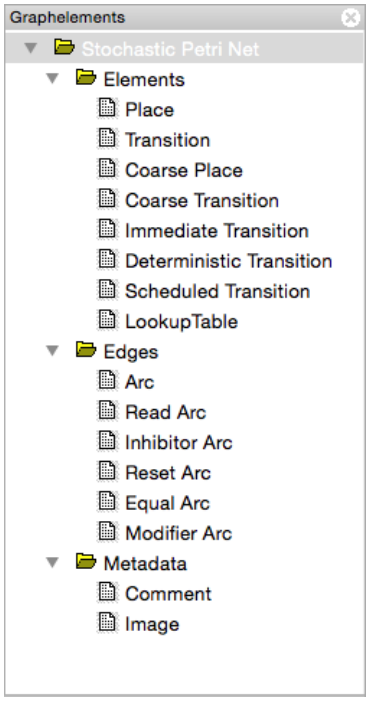


2 Graph Elements **3**

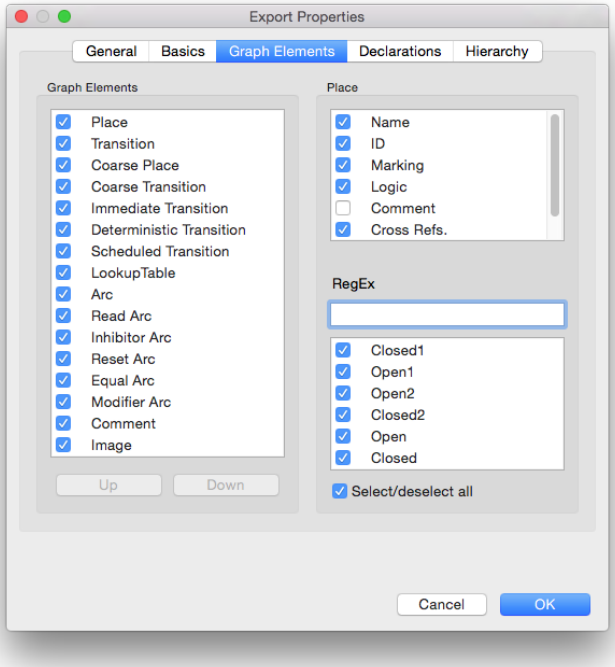
2.1 Places . . . 3
2.2 Transitions . . . 17
2.3 Coarse Places . . . 22
2.4 Coarse Transitions . . . 23
2.5 Arcs . . . 24
2.6 Comments . . . 46
2.7 Images . . . 47

(c) Generated Report: Graph Elements for Petri Net

Figure A.24: Exported graph Elements for Petri Net.



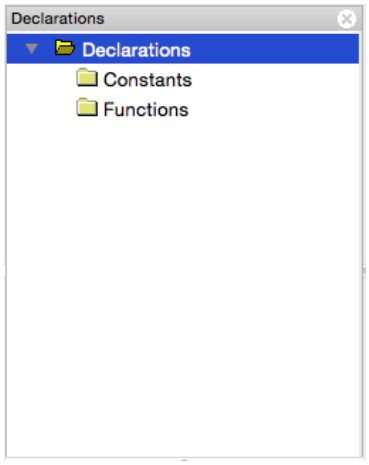
(a) Snoopy: Graph Elements Panel



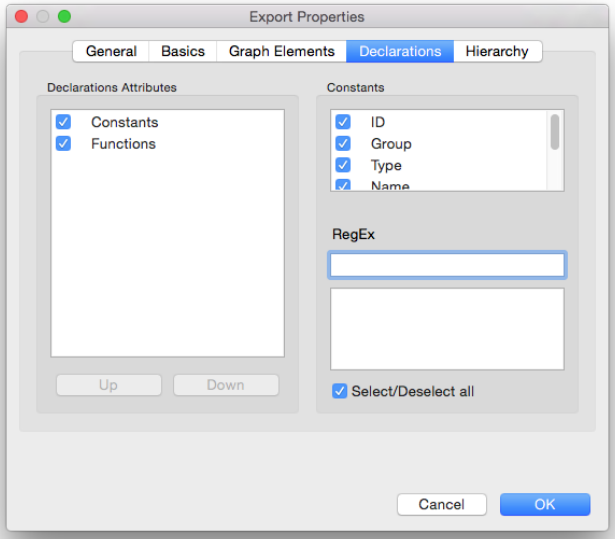
(b) Snoopy2LaTeX: Graph Elements for Stochastic Petri Net

(c) Generated Report: Graph Elements for Stochastic Petri Net

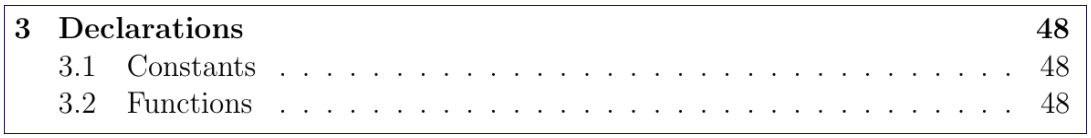
Figure A.25: Exported Graph Elements for Stochastic Petri Net.



(a) Snoopy: Declarations Panel

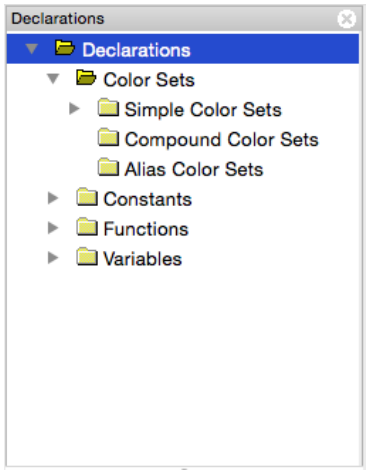


(b) Snoopy2LaTeX: Declarations for Petri Net

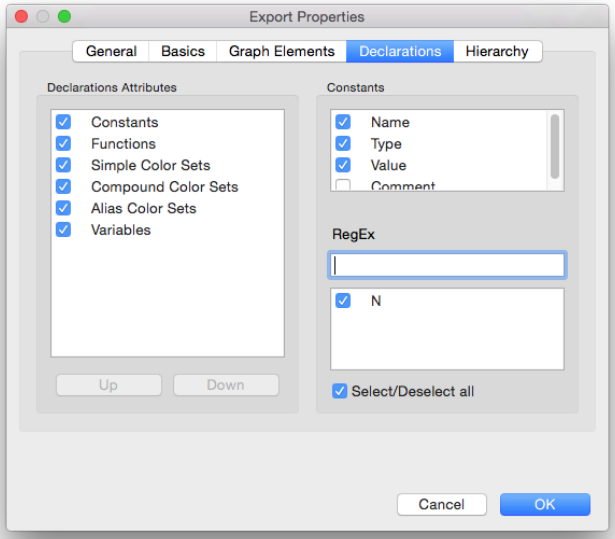


(c) Generated Report: Declarations for Petri Net

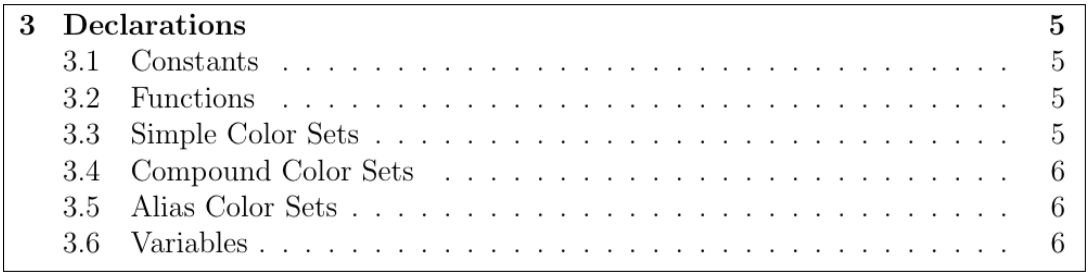
Figure A.26: Exported Declarations for Petri Net.



(a) Snoopy: Declarations Panel



(b) Snoopy2LaTeX: Declarations for Colored Petri Net

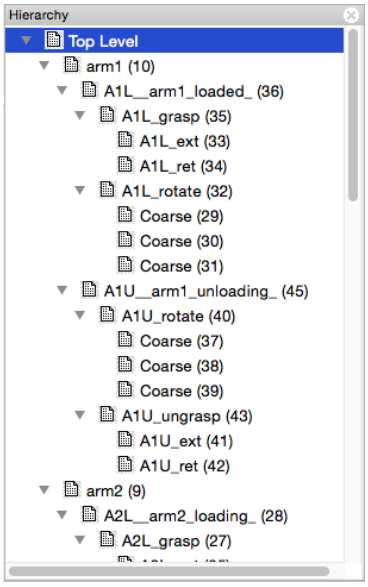


3 Declarations **5**

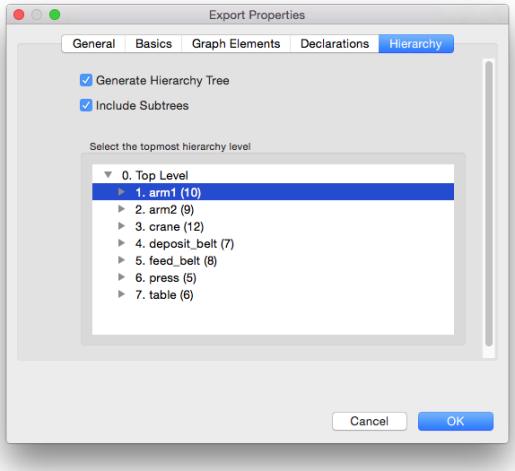
3.1 Constants 5
3.2 Functions 5
3.3 Simple Color Sets 5
3.4 Compound Color Sets 6
3.5 Alias Color Sets 6
3.6 Variables 6

(c) Generated Report: Declarations for Colored Petri Net

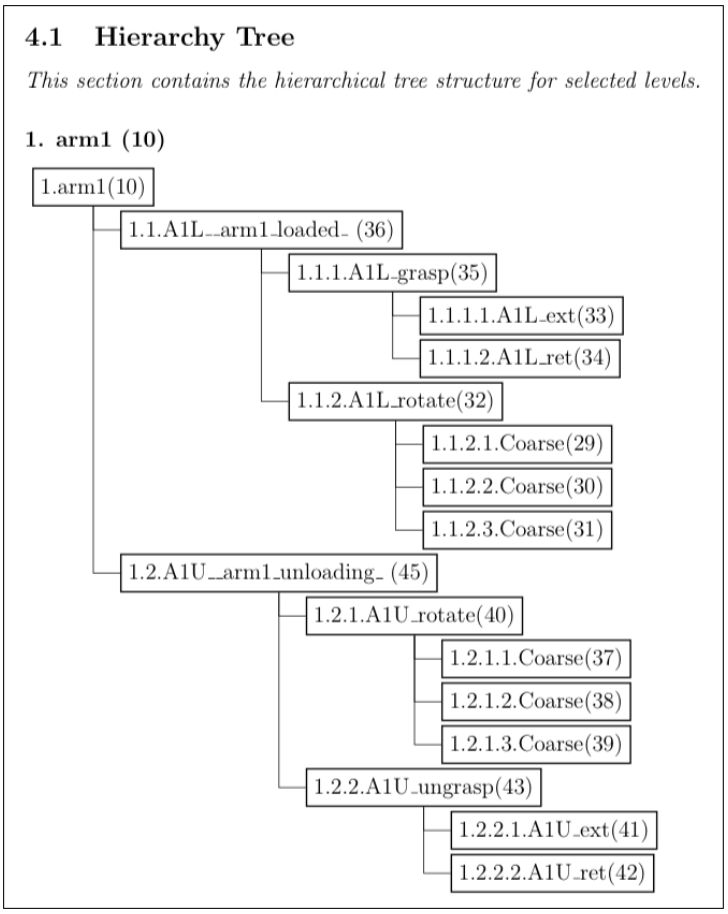
Figure A.27: Exported Declarations for Colored Petri Net.



(a) Snoopy: Hierarchy Panel

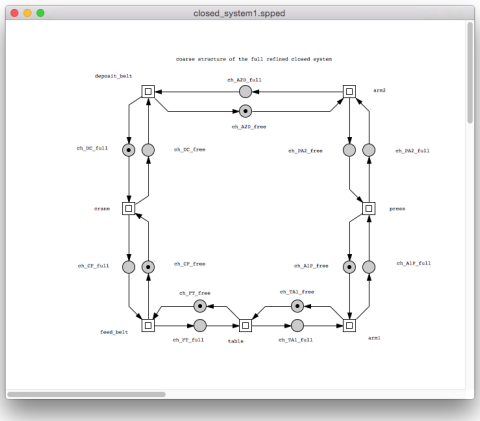


(b) Snoopy2LaTeX: Hierarchy Selection

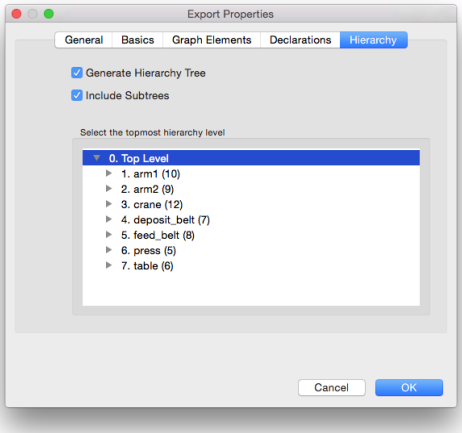


(c) Generated Report: Hierarchy Tree

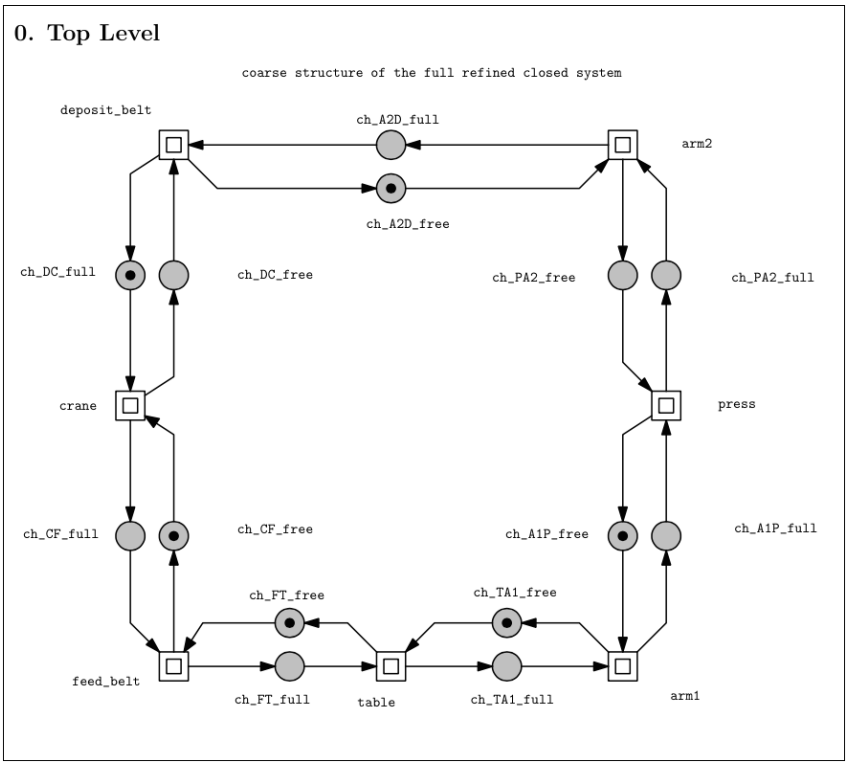
Figure A.28: Exported Hierarchy Tree for a selected level.



(a) Petri Net in Snoopy

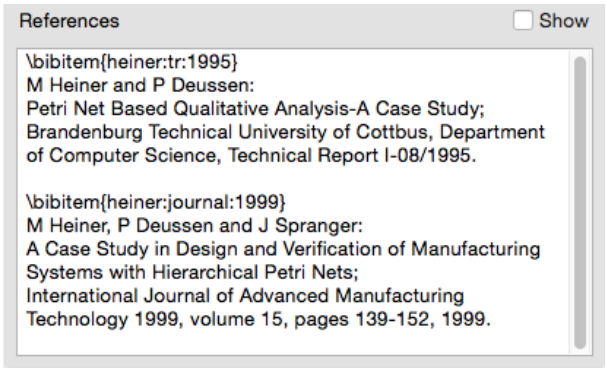


(b) Snoopy2LaTeX: Hierarchy Selection



(c) Generated Report: Hierarchy Tree

Figure A.29: Exported Hierarchy Figure for a selected level.



(a) References in Snoopy

↓

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

[3] M Heiner and P Deussen: Petri Net Based Qualitative Analysis-A Case Study; Brandenburg Technical University of Cottbus, Department of Computer Science, Technical Report I-08/1995.

[4] M Heiner, P Deussen and J Spranger: A Case Study in Design and Verification of Manufacturing Systems with Hierarchical Petri Nets; International Journal of Advanced Manufacturing Technology 1999, volume 15, pages 139-152, 1999.

(b) Exported References

Figure A.30: Exported References for a given net.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

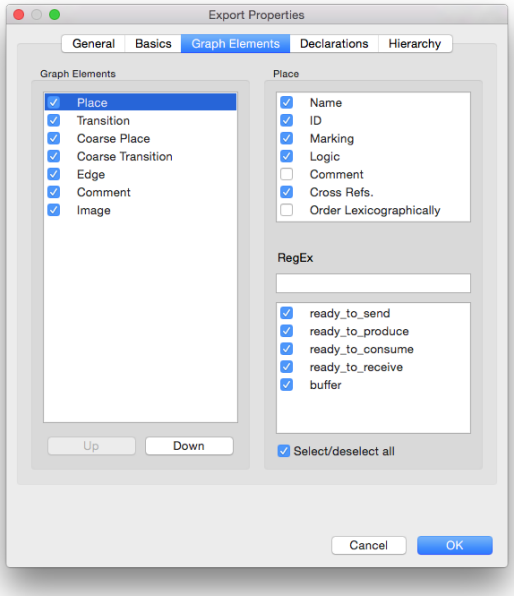
Figure A.31: Exported Glossary

- **Element List in tabs correspond to tables in report (nodes as tuples)**

  For *Graph Elements* and *Declarations*, each sub-section (corresponding to element type) consists of exported information for all nodes of that type. This information is presented as **tables**. Each element type has a table with:

  1. **Nodes** selected for export as the rows of table
  2. **Attributes** selected for export as the columns of table

  Figure A.32 shows a table generated for 'Places' graph element, listing all the node of place type as tuples entries. Similarly, Figure A.33 shows how edges are exported to corresponding table entries.



(a) Snoop2LaTeX: Custom Places

↓

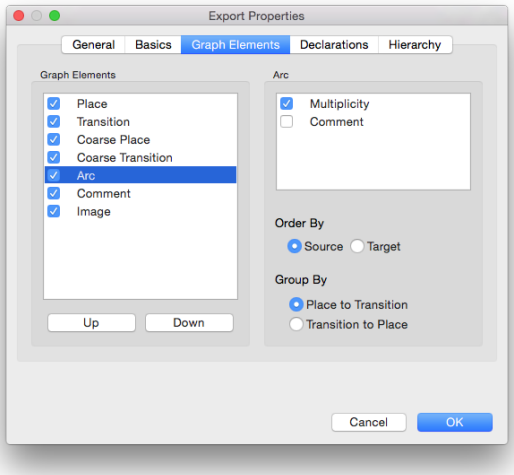
2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ready_to_send	0	0	✗	producer(2)
ready_to_produce	1	1	✗	producer(2)
ready_to_consume	2	0	✗	consumer(3)
ready_to_receive	3	1	✗	consumer(3)
buffer	4	0	✗	TopLevel producer(2) consumer(3)

(b) Generates Report: Places Table

Figure A.32: Exported Places for a given net.



(a) Snoop2LaTeX: Custom Arcs

↓

2.5 Arcs

Table 4: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
buffer	P	receive	T	1
ready_to_consume	P	consume	T	1
ready_to_produce	P	produce	T	1
ready_to_receive	P	receive	T	1
ready_to_send	P	send	T	1
consume	T	ready_to_receive	P	1
produce	T	ready_to_send	P	1
receive	T	ready_to_consume	P	1
send	T	buffer	P	1
send	T	ready_to_produce	P	1

(b) Generates Report: Arcs Table

Figure A.33: Exported Arcs for a given net.

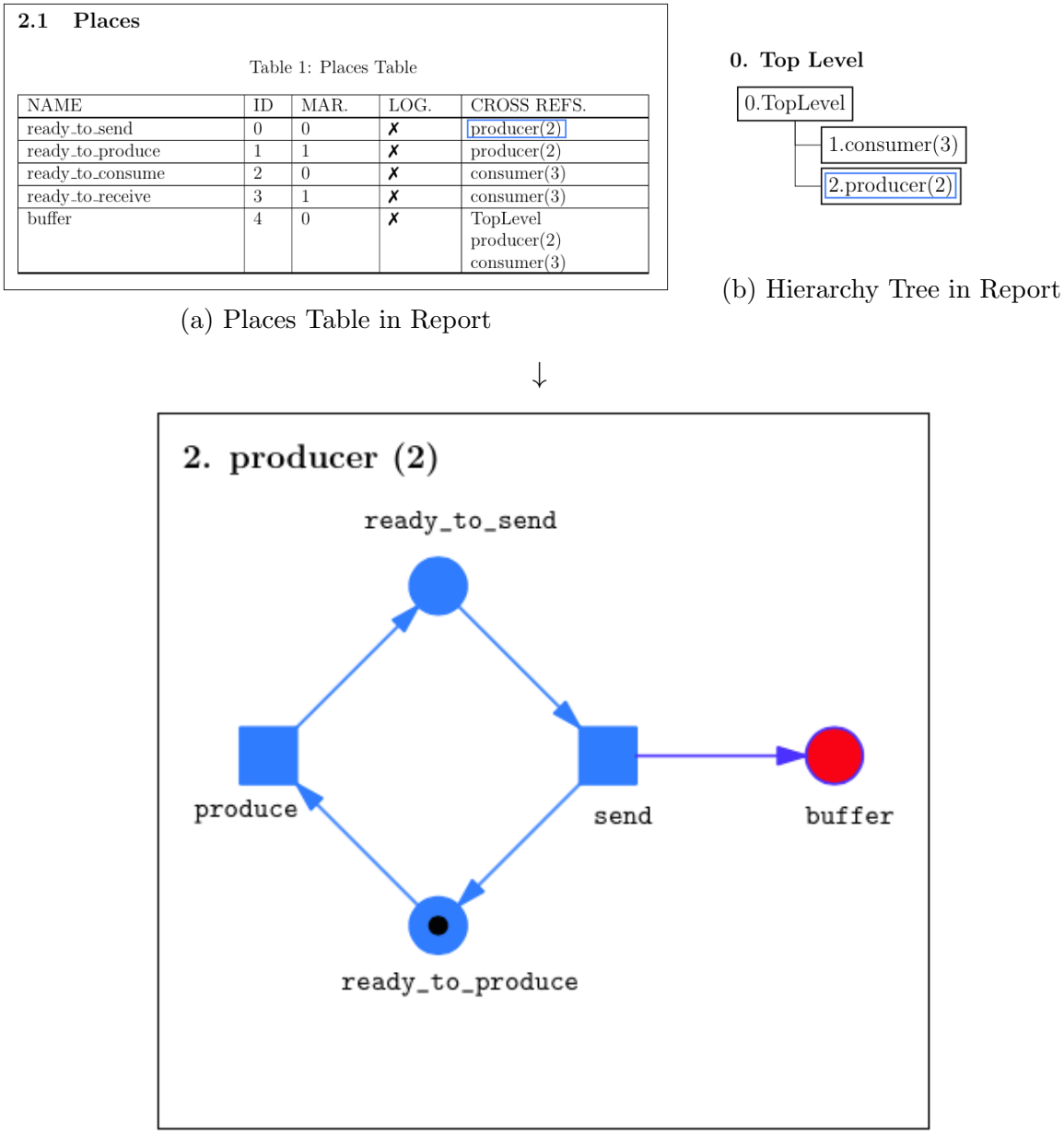
- **Extensive cross-referencing**

  The generated report consists of efficient cross-referencing for all its elements.

  1. Each node name in **Arcs Table** links to the tuple with its description. For instance, if the node is a Coarse Place, then it links to the tuple for this node in the 'Coarse Places Table'.
  2. Each net level label under **CROSS REFS.** column for any element table links to its figure in the 'Hierarchy Figures' subsection (illustrated in Figure A.34).
  3. Each node in the **Hierarchy Tree** also links to its corresponding figure in the 'Hierarchy Figures' subsection (illustrated in Figure A.34).
  4. If **Images** are present in a net and are exported, then thumbnail for each

exported image in the **Image Table** links to its enlarged version in the appendix.

5. By the default property of LaTeX, every every in the table of contents for the report links to its corresponding section.



(a) Places Table in Report

(b) Hierarchy Tree in Report

↓

(c) Net figure for a given net level

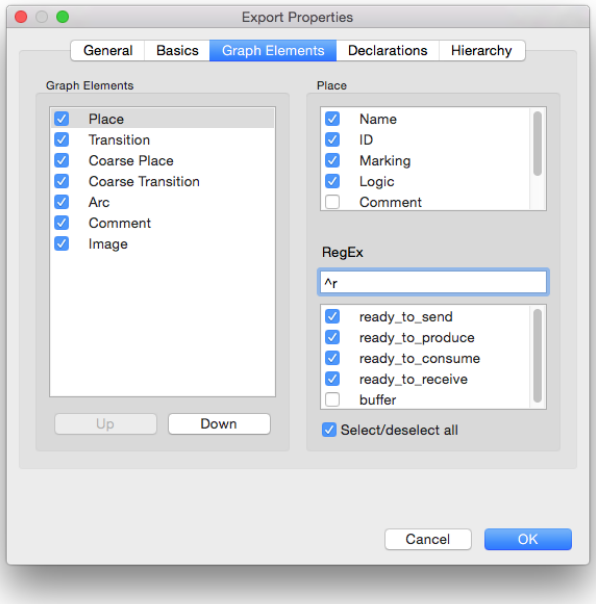
Figure A.34: Cross-referencing Illustration. In Figure (a) and (b), the level label enclosed in a blue rectangle indicates a cross-reference. This when clicked, leads to the corresponding net figure as shown in Figure (c). All net levels have similar cross-referencing in the generated report.

A.6 Using Regular Expressions

This section explains the use of regular expressions (abbreviated as **regex**) in our Export dialog and the basic terminology for using them.

A.6.1 Usage in Snoopy2LaTeX

In the Snoopy2LaTeX Export dialog, the *Graph Elements* and *Declarations* tab have a 'RegEx Field' as shown in Figure A.15 and A.17, respectively. This text field allows the user to enter a regular expression for the desired set of strings (node names in the respective node checklist). The use of regex facilitates a compact way of describing the sets of nodes which confirm to a pattern. This offers huge significance in case of large Petri nets with thousands of nodes, where manual selection of each node individually can be tedious.



(a) Export dialog with Regex for Node Selection

↓

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ready_to_send	0	0	✗	producer(2)
ready_to_produce	1	1	✗	producer(2)
ready_to_consume	2	0	✗	consumer(3)
ready_to_receive	3	1	✗	consumer(3)

(b) Generated Node List in Report

Figure A.35: Illustration for use of Regex in Snoopy2LaTeX

As shown in Figure A.35(a), once the user enters a regex (e.g. '^r') in the regex field, the corresponding subset of node list gets selected (e.g. Here in figure, all nodes with names starting with 'r' get selected). On export, only this subset of nodes gets exported (as shown in Figure A.35(b)).

A.6.2 Basic Terminology

The Regex library used for our Snoopy2LaTeX Export provides support for **POSIX Extended Regular Expressions (ERE)** syntax. This syntax is *case-sensitive*. Table A.2 lists the various metacharacters for POSIX regex and their description.

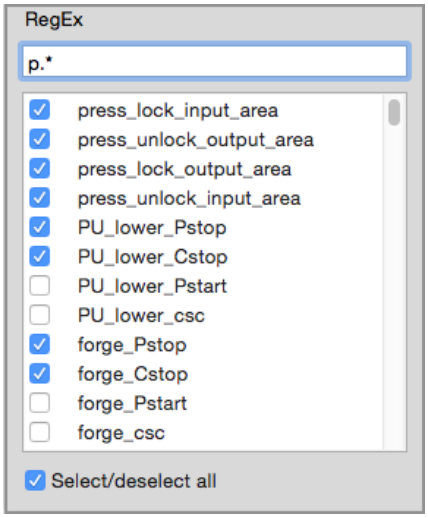
Metacharacter	Description
`.`	Matches any single character. Within POSIX bracket expressions, the dot character matches a literal dot. For example, `a.c` matches "abc", etc., but `[a.c]` matches only "a", ".", or "c".
`[ ]`	A bracket expression. Matches a single character that is contained within the brackets. For example, `[abc]` matches "a", "b", or "c". `[a-z]` specifies a range which matches any lowercase letter from "a" to "z". These forms can be mixed: `[abcx-z]` matches "a", "b", "c", "x", "y", or "z", as does `[a-cx-z]` . The `-` character is treated as a literal character if it is the last or the first (after the ^, if present) character within the brackets: `[abc-]`, `[-abc]`. Note that backslash escapes are not allowed. The `]` character can be included in a bracket expression if it is the first (after the ^) character: `[]abc]`.
`[ ^ ]`	Matches a single character that is not contained within the brackets. For example, `[ ^abc]` matches any character other than"a", "b", or "c". `[ ^a-z]` matches any single character that is not a lowercase letter from "a" to "z". Likewise, literal characters and ranges can be mixed.
`^`	Matches the starting position within the string. In line-based tools, it matches the starting position of any line.
`$`	Matches the ending position of the string or the position just before a string-ending newline. In line-based tools, it matches the ending position of any line.
`( )`	Defines a marked subexpression. The string matched within the parentheses can be recalled later (see the next entry, `\n`). A marked subexpression is also called a block or capturing group.
`\n`	Matches what the nth marked subexpression matched, where n is a digit from 1 to 9. This construct is vaguely defined in the POSIX.2 standard. Some tools allow referencing more than nine capturing groups.
`*`	Matches the preceding element zero or more times. For example, ab*c matches "ac", "abc", "abbbc", etc. `[xyz]*` matches "", "x", "y", "z", "zx", "zyx", "xyzzy", and so on. `(ab)*` matches "", "ab", "abab", "ababab", and so on.
`{m,n}`	Matches the preceding element at least m and not more than n times. For example, `a{3,5}` matches only "aaa", "aaaa", and "aaaaa". This is not found in a few older instances of regular expressions.
`?`	Matches the preceding element zero or one time. For example, `ab?c` matches only "ac" or "abc".
`+`	Matches the preceding element one or more times. For example, `ab+c` matches "abc", "abbc", "abbbc", and so on, but not "ac".
`|`	The choice (also known as alternation or set union) operator matches either the expression before or the expression after the operator. For example, abc|def matches "abc" or "def".

Table A.2: Syntax for POSIX Extended Regular Expressions. Taken from [4].

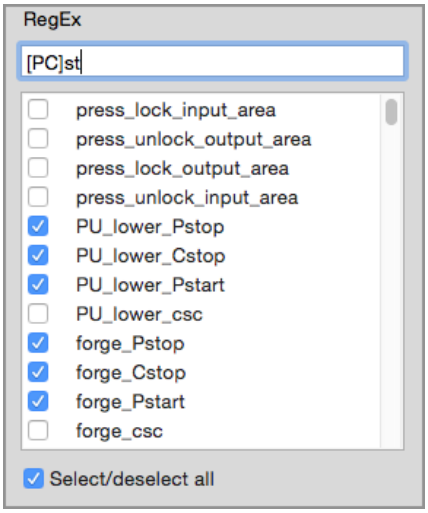
Some general regex examples:

- `.at` matches any three-character string ending with "at", including "hat", "cat", and "bat".
- `[hc]at` matches "hat" and "cat".
- `[^b]at` matches all strings matched by `.at` except "bat".
- `[^hc]at` matches all strings matched by `.at` other than "hat" and "cat".
- `^[hc]at` matches "hat" and "cat", but only at the beginning of the string or line.
- `[hc]at$` matches "hat" and "cat", but only at the end of the string or line.
- `\[.\]` matches any single character surrounded by "[" and "]" since the brackets are escaped, for example: "[a]" and "[b]".
- `s.*` matches any number of characters preceded by `s`, for example: "saw" and "seed".
- `[hc]+at` matches "hat", "cat", "hhat", "chat", "hcat", "cchchat", and so on, but not "at".
- `[hc]?at` matches "hat", "cat", and "at".
- `[hc]*at` matches "hat", "cat", "hhat", "chat", "hcat", "cchchat", "at", and so on.
- `cat|dog` matches "cat" or "dog".

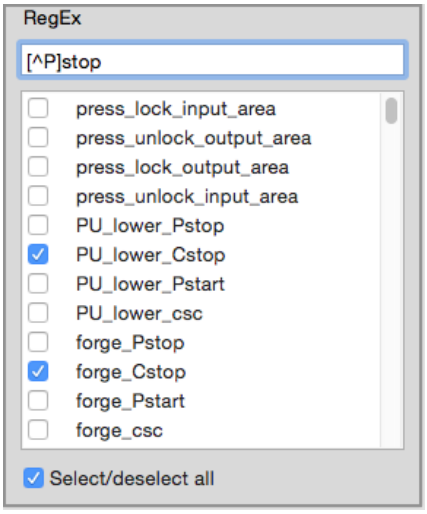
Some examples for use of regex in Snoopy2LaTeX are illustrated in figure A.36. For every entry of a *valid regular expression* in the 'Regex' field, the matching set of node names get selected in the node checklist.



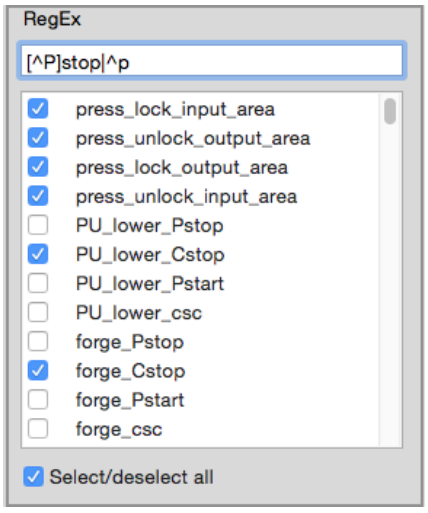
(a) Regex: `p.*`



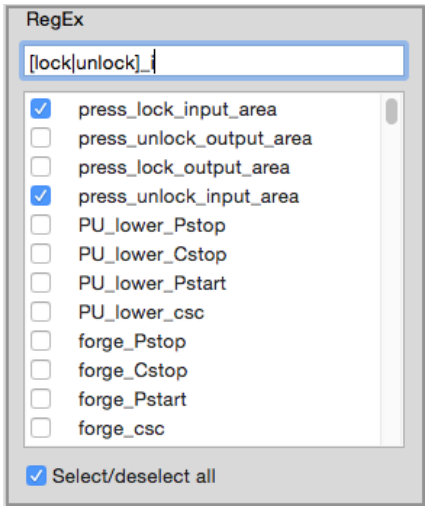
(b) Regex: `[PC]st`



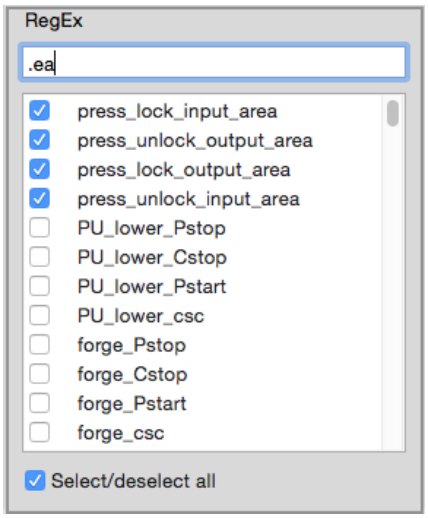
(c) Regex: `[^P]stop`



(d) Regex: `[^P]stop|^p`



(e) Regex: `[lock|unlock]_i`



(f) Regex: `.ea`

Figure A.36: Regex in Snoopy2LaTeX. As the user enters a valid regex in the 'RegEx' field of Export dialog, the matching set of nodes gets selected in the respective node checklist.

A.7 Case Studies

This section presents case studies for all the graph classes in Snoopy.

A.7.1 Qualitative Nets

A.7.1.1 Closed_system.pn (Petri Net)

This is a large Petri net with a complex hierarchy. The top level for this Snoopy file is shown in Figure A.37. Customizations done in Snoopy2LaTeX dialog:

- **Places** selected again regex - "ch"
- **Transitions** selected again regex - ".lock"
- **Hierarchy** selected as shown Figure A.38

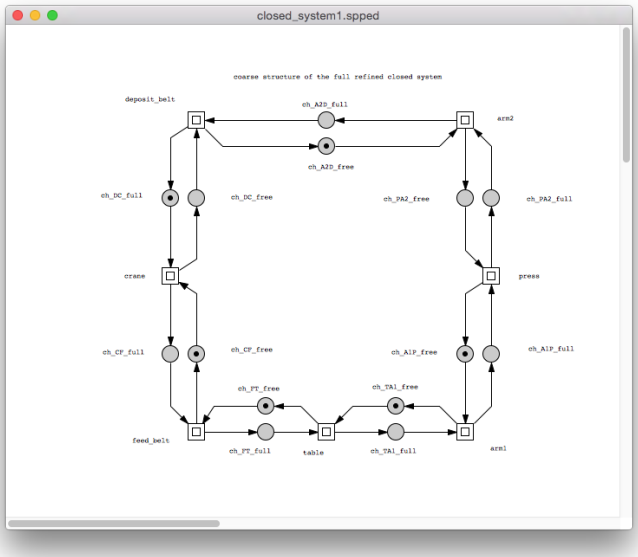


Figure A.37: Production Cell - Snoopy file

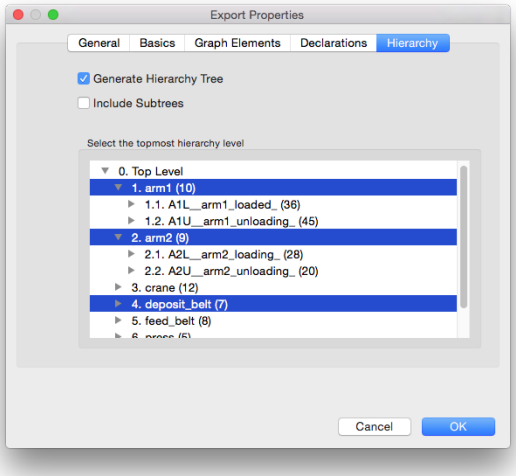
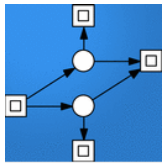


Figure A.38: Custom Hierarchy in Snoopy2LaTeX

The following pages show the generated PDF report.



closed_system.spped

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

26/05/2015

closed_system.spped Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: closed_system.spped
Authors: A. Sharma, M. Heiner
Creation Timestamp: 2015-01-22 15:24:03
Keywords: Petri net, closed system, full refined closed system
References: [3, 4]

1.2 Net Informations

Net Class: Petri Net

Element	Count
Place	231
Transition	202
Edge	846

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ch_DC_full	0	1	✓	TopLevel deposit_belt(7) crane(12)
ch_DC_free	1	0	✓	TopLevel deposit_belt(7) crane(12)
ch_CF_free	2	1	✓	TopLevel feed_belt(8) crane(12)
ch_CF_full	3	0	✓	TopLevel feed_belt(8) crane(12)
ch_A1P_full	4	0	✓	TopLevel press(5) arm1(10)
ch_A1P_free	5	1	✓	TopLevel press(5) arm1(10)
ch_TA1_full	6	0	✓	TopLevel table(6) arm1(10)
ch_TA1_free	7	1	✓	TopLevel table(6) arm1(10)
ch_A2D_full	8	0	✓	TopLevel deposit_belt(7) arm2(9)
ch_A2D_free	9	1	✓	TopLevel deposit_belt(7) arm2(9)
ch_FT_free	10	1	✓	TopLevel table(6) feed_belt(8)
ch_FT_full	11	0	✓	TopLevel table(6) feed_belt(8)

Continued on next page

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
ch_PA2_free	12	0	✓	TopLevel press(5) arm2(9)
ch_PA2_full	13	0	✓	TopLevel press(5) arm2(9)

2.2 Transitions

Table 2: Transitions Table

NAME	ID	LOG.	CROSS REFS.
press_lock_input_area	0	✗	press(5) press_go_unload_pos__PU_ (49)
press_unlock_output_area	1	✗	press(5) press_go_unload_pos__PU_ (49)
press_lock_output_area	2	✗	press(5) press_go_load_pos__PL_ (50)
press_unlock_input_area	3	✗	press(5) press_go_load_pos__PL_ (50)
table_lock_input_area	16	✗	table(6) table_go_unload_pos(51)
table_unlock_output_area	17	✗	table(6) table_go_unload_pos(51)
table_lock_output_area	18	✗	table(6) table_go_load_pos(54)
table_unlock_input_area	19	✗	table(6) table_go_load_pos(54)
deposit_belt_lock_input_area	36	✗	deposit_belt(7)
deposit_belt_lock_output_area	37	✗	deposit_belt(7) deposit_belt_transporting(60)
deposit_belt_unlock_input_area	38	✗	deposit_belt(7) deposit_belt_transporting(60)
deposit_belt_unlock_output_area	39	✗	deposit_belt(7)
feed_belt_lock_input_area	48	✗	feed_belt(8)
feed_belt_lock_output_area	49	✗	feed_belt(8) feed_belt_transporting(57)
feed_belt_unlock_input_area	50	✗	feed_belt(8) feed_belt_transporting(57)

Continued on next page

Table 2: Transitions Table

NAME	ID	LOG.	CROSS REFS.
feed_belt_unlock_output_area	51	✗	feed_belt(8)
arm2_lock_input_area	60	✗	arm2(9)
arm2_lock_swivel_1	61	✗	arm2(9) A2L__arm2_loading_ (28)
arm2_unlock_swivel_1	62	✗	arm2(9)
arm2_lock_output_area	63	✗	arm2(9)
arm2_unlock_output_area	64	✗	arm2(9) A2U__arm2_unloading_ (20)
arm2_unlock_swivel_2	65	✗	arm2(9)
arm2_unlock_input_area	66	✗	arm2(9) A2L__arm2_loading_ (28)
arm2_lock_swivel_2	67	✗	arm2(9) A2U__arm2_unloading_ (20)
arm1_lock_input_area	116	✗	arm1(10)
arm1_lock_swivel_1	117	✗	arm1(10) A1L__arm1_loaded_ (36)
arm1_unlock_swivel_1	118	✗	arm1(10)
arm1_lock_output_area	119	✗	arm1(10)
arm1_unlock_output_area	120	✗	arm1(10) A1U__arm1_unloading_ (45)
arm1_unlock_swivel_2	121	✗	arm1(10)
arm1_unlock_input_area	122	✗	arm1(10) A1L__arm1_loaded_ (36)
arm1_lock_swivel_2	123	✗	arm1(10) A1U__arm1_unloading_ (45)
crane_lock_input_area	172	✗	crane(12) crane_loading(67)
crane_lock_output_area	173	✗	crane(12) crane_unloading(63)
crane_unlock_output_area	174	✗	crane(12) crane_unloading(63)
crane_unlock_input_area	175	✗	crane(12) crane_loading(67)

2.3 Coarse Places

Table 3: Coarse Places Table

NAME	NET.	CROSS REFS.
press_go_unload_pos__PU_	5	press(5)
press_go_load_pos__PL_	5	press(5)
table_go_load_pos	6	table(6)
table_go_unload_pos	6	table(6)
deposit_belt_transporting	7	deposit_belt(7)
feed_belt_transporting	8	feed_belt(8)
A2U__arm2_unloading_	9	arm2(9)
A2L__arm2_loading_	9	arm2(9)
A1L__arm1_loaded_	10	arm1(10)
A1U__arm1_unloading_	10	arm1(10)
crane_unloading	12	crane(12)
crane_loading	12	crane(12)

2.4 Coarse Transitions

Table 4: Coarse Transitions Table

NAME	NET.	CROSS REFS.
press	1	TopLevel
table	1	TopLevel
deposit_belt	1	TopLevel
feed_belt	1	TopLevel
arm2	1	TopLevel
arm1	1	TopLevel
crane	1	TopLevel
lower	49	press_go_unload_pos__PU_ (49)
forge	49	press_go_unload_pos__PU_ (49)
lower	49	press_go_unload_pos__PU_ (49)
rotate	54	table_go_load_pos(54)
lower	49	press_go_unload_pos__PU_ (49)
lift	51	table_go_unload_pos(51)
rotate	54	table_go_load_pos(54)
trans	60	deposit_belt_transporting(60)
deliver	60	deposit_belt_transporting(60)
trans	60	deposit_belt_transporting(60)
deliver	60	deposit_belt_transporting(60)

Continued on next page

Table 4: Coarse Transitions Table

NAME	NET.	CROSS REFS.
A2U_rotate	20	A2U__arm2_unloading_ (20)
A2U_ungrasp	19	A2U_ungrasp(19)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
A2U_ext	19	A2U_ungrasp(19)
A2U_ret	19	A2U_ungrasp(19)
A2L_rotate	28	A2L__arm2_loading_ (28)
A2L_grasp	27	A2L_grasp(27)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
A2L_ext	27	A2L_grasp(27)
A2L_ret	27	A2L_grasp(27)
A1L_rotate	36	A1L__arm1_loaded_ (36)
A1L_grasp	35	A1L_grasp(35)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
A1L_ext	35	A1L_grasp(35)
A1L_ret	35	A1L_grasp(35)
A1U_rotate	45	A1U__arm1_unloading_ (45)
A1U_ungrasp	43	A1U_ungrasp(43)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
	16	A2U_rotate(16)
A1U_ext	43	A1U_ungrasp(43)
A1U_ret	43	A1U_ungrasp(43)
lift	51	table_go_unload_pos(51)
transport	63	crane_unloading(63)
lower	49	press_go_unload_pos__PU_ (49)
lower	49	press_go_unload_pos__PU_ (49)
transport	63	crane_unloading(63)
lift	51	table_go_unload_pos(51)

2.5 Edges

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A1L_ext_rs	P	A1L_ext_Pstop	T	1
A1L_ext_run	P	A1L_ext_Cstop	T	1
A1L_ext_run	P	A1L_ext_csc	T	1
A1L_in	P	A1L_c1	T	1
A1L_in	P	A1L_c2	T	1
A1L_in	P	A1L_c3	T	1
A1L_loaded	P	A1L_ret_Pstart	T	1
A1L_out	P	arm1_unlock_input_area	T	1
A1L_ret_rs	P	A1L_ret_Pstop	T	1
A1L_ret_run	P	A1L_ret_Cstop	T	1
A1L_ret_run	P	A1L_ret_csc	T	1
A1L_rot1_in	P	A1L_rot1_Pstart	T	1
A1L_rot1_rs	P	A1L_rot1_Pstop	T	1
A1L_rot1_run	P	A1L_rot1_Cstop	T	1
A1L_rot1_run	P	A1L_rot1_csc	T	1
A1L_rot2_in	P	A1L_rot2_Pstart	T	1
A1L_rot2_rs	P	A1L_rot2_Pstop	T	1
A1L_rot2_run	P	A1L_rot2_Cstop	T	1
A1L_rot2_run	P	A1L_rot2_csc	T	1
A1L_rot3_in	P	A1L_rot3_Pstart	T	1
A1L_rot3_rs	P	A1L_rot3_Pstop	T	1
A1L_rot3_run	P	A1L_rot3_Cstop	T	1
A1L_rot3_run	P	A1L_rot3_csc	T	1
A1L_rotated	P	A1L_ext_Pstart	T	1
A1U_ext_rs	P	A1U_ext_Pstop	T	1
A1U_ext_run	P	A1U_ext_Cstop	T	1
A1U_ext_run	P	A1U_ext_csc	T	1
A1U_extendet	P	A1U_ungrasp	T	1
A1U_in	P	A1U_c1	T	1
A1U_in	P	A1U_c2	T	1
A1U_in	P	A1U_c3	T	1
A1U_out	P	arm1_unlock_output_area	T	1
A1U_ret_rs	P	A1U_ret_Pstop	T	1
A1U_ret_run	P	A1U_ret_Cstop	T	1
A1U_ret_run	P	A1U_ret_csc	T	1
A1U_rot1_in	P	A1U_rot1_Pstart	T	1
A1U_rot1_rs	P	A1U_rot1_Pstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A1U_rot1_run	P	A1U_rot1_Cstop	T	1
A1U_rot1_run	P	A1U_rot1_csc	T	1
A1U_rot2_in	P	A1U_rot2_Pstart	T	1
A1U_rot2_rs	P	A1U_rot2_Pstop	T	1
A1U_rot2_run	P	A1U_rot2_Cstop	T	1
A1U_rot2_run	P	A1U_rot2_csc	T	1
A1U_rot3_in	P	A1U_rot3_Pstart	T	1
A1U_rot3_rs	P	A1U_rot3_Pstop	T	1
A1U_rot3_run	P	A1U_rot3_Cstop	T	1
A1U_rot3_run	P	A1U_rot3_csc	T	1
A1U_rotated	P	A1U_ext_Pstart	T	1
A1U_unloadet	P	A1U_ret_Pstart	T	1
A1_extended	P	A1L_grasp	T	1
A2L_ext_rs	P	A2L_ext_Pstop	T	1
A2L_ext_run	P	A2L_ext_Cstop	T	1
A2L_ext_run	P	A2L_ext_csc	T	1
A2L_extended	P	A2L_grasp	T	1
A2L_in	P	A2L_c1	T	1
A2L_in	P	A2L_c2	T	1
A2L_in	P	A2L_c3	T	1
A2L_loaded	P	A2L_ret_Pstart	T	1
A2L_out	P	arm2_unlock_input_area	T	1
A2L_ret_rs	P	A2L_ret_Pstop	T	1
A2L_ret_run	P	A2L_ret_Cstop	T	1
A2L_ret_run	P	A2L_ret_csc	T	1
A2L_rot1_in	P	A2L_rot1_Pstart	T	1
A2L_rot1_rs	P	A2L_rot1_Pstop	T	1
A2L_rot1_run	P	A2L_rot1_Cstop	T	1
A2L_rot1_run	P	A2L_rot1_csc	T	1
A2L_rot2_in	P	A2L_rot2_Pstart	T	1
A2L_rot2_rs	P	A2L_rot2_Pstop	T	1
A2L_rot2_run	P	A2L_rot2_Cstop	T	1
A2L_rot2_run	P	A2L_rot2_csc	T	1
A2L_rot3_in	P	A2L_rot3_Pstart	T	1
A2L_rot3_rs	P	A2L_rot3_Pstop	T	1
A2L_rot3_run	P	A2L_rot3_Cstop	T	1
A2L_rot3_run	P	A2L_rot3_csc	T	1
A2L_rotated	P	A2L_ext_Pstart	T	1
A2U_ext_rs	P	A2U_ext_Pstop	T	1
A2U_ext_run	P	A2U_ext_Cstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A2U_ext_run	P	A2U_ext_csc	T	1
A2U_extended	P	A2U_ungrasp	T	1
A2U_in	P	A2U_c1	T	1
A2U_in	P	A2U_c2	T	1
A2U_in	P	A2U_c3	T	1
A2U_out	P	arm2_unlock_output_area	T	1
A2U_ret_rs	P	A2U_ret_Pstop	T	1
A2U_ret_run	P	A2U_ret_Cstop	T	1
A2U_ret_run	P	A2U_ret_csc	T	1
A2U_rot1_in	P	A2U_rot1_Pstart	T	1
A2U_rot1_rs	P	A2U_rot1_Pstop	T	1
A2U_rot1_run	P	A2U_rot1_Cstop	T	1
A2U_rot1_run	P	A2U_rot1_csc	T	1
A2U_rot2_in	P	A2U_rot2_Pstart	T	1
A2U_rot2_rs	P	A2U_rot2_Pstop	T	1
A2U_rot2_run	P	A2U_rot2_Cstop	T	1
A2U_rot2_run	P	A2U_rot2_csc	T	1
A2U_rot3_in	P	A2U_rot3_Pstart	T	1
A2U_rot3_rs	P	A2U_rot3_Pstop	T	1
A2U_rot3_run	P	A2U_rot3_Cstop	T	1
A2U_rot3_run	P	A2U_rot3_csc	T	1
A2U_rotated	P	A2U_ext_Pstart	T	1
A2U_unloaded	P	A2U_ret_Pstart	T	1
CL_in	P	CL_lower_Pstart	T	1
CL_lift_rs	P	CL_lift_Pstop	T	1
CL_lift_run	P	CL_lift_Cstop	T	1
CL_lift_run	P	CL_lift_csc	T	1
CL_loaded	P	CL_lift_Pstart	T	1
CL_lower_rs	P	CL_lower_Pstop	T	1
CL_lower_run	P	CL_lower_Cstop	T	1
CL_lower_run	P	CL_lower_csc	T	1
CL_out	P	crane_unlock_input_area	T	1
CL_ready_to_grasp	P	CL_grasp	T	1
CL_ready_to_transport	P	CL_trans_Pstart	T	1
CL_trans_rs	P	CL_trans_Pstop	T	1
CL_trans_run	P	CL_trans_Cstop	T	1
CL_trans_run	P	CL_trans_csc	T	1
CU_in	P	CU_lower_Pstart	T	1
CU_lift_rs	P	CU_lift_Pstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
CU_lift_run	P	CU_lift_Cstop	T	1
CU_lift_run	P	CU_lift_csc	T	1
CU_lower_rs	P	CU_lower_Pstop	T	1
CU_lower_run	P	CU_lower_Cstop	T	1
CU_lower_run	P	CU_lower_csc	T	1
CU_out	P	crane_unlock_output_area	T	1
CU_ready_to_transport	P	CU_trans_Pstart	T	1
CU_ready_to_ungrasp	P	CU_ungrasp	T	1
CU_trans_rs	P	CU_trans_Pstop	T	1
CU_trans_run	P	CU_trans_Cstop	T	1
CU_trans_run	P	CU_trans_csc	T	1
CU_unloaded	P	CU_lift_Pstart	T	1
DB_at_end	P	DB_deliver_Pstart	T	1
DB_deliver_rs	P	DB_deliver_Pstop	T	1
DB_deliver_run	P	DB_deliver_Cstop	T	1
DB_deliver_run	P	DB_deliver_csc	T	1
DB_in	P	DB_trans_Pstart	T	1
DB_out	P	deposit_belt_unlock_input_area	T	1
DB_trans_rs	P	DB_trans_Pstop	T	1
DB_trans_run	P	DB_trans_Cstop	T	1
DB_trans_run	P	DB_trans_csc	T	1
FB_at_end	P	FB_deliver_Pstart	T	1
FB_deliver_rs	P	FB_deliver_Pstop	T	1
FB_deliver_run	P	FB_deliver_Cstop	T	1
FB_deliver_run	P	FB_deliver_csc	T	1
FB_in	P	FB_trans_Pstart	T	1
FB_out	P	feed_belt_unlock_input_area	T	1
FB_trans_rs	P	FB_trans_Pstop	T	1
FB_trans_run	P	FB_trans_Cstop	T	1
FB_trans_run	P	FB_trans_csc	T	1
PL_in	P	PL_lower_Pstart	T	1
PL_lower_rs	P	PL_lower_Pstop	T	1
PL_lower_run	P	PL_lower_Cstop	T	1
PL_lower_run	P	PL_lower_csc	T	1
PL_out	P	press_unlock_input_area	T	1
PU_in	P	forge_Pstart	T	1
PU_lower_rs	P	PU_lower_Pstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
PU_lower_run	P	PU_lower_Cstop	T	1
PU_lower_run	P	PU_lower_csc	T	1
PU_out	P	press_unlock_output_area	T	1
TL_in	P	TL_rot_Pstart	T	1
TL_lower_rs	P	TL_lower_Pstop	T	1
TL_lower_run	P	TL_lower_Cstop	T	1
TL_lower_run	P	TL_lower_csc	T	1
TL_out	P	table_unlock_input_area	T	1
TL_rot_rs	P	TL_rot_Pstop	T	1
TL_rot_run	P	TL_rot_Cstop	T	1
TL_rot_run	P	TL_rot_csc	T	1
TU_in	P	TU_rot_Pstart	T	1
TU_lift_rs	P	TU_lift_Pstop	T	1
TU_lift_run	P	TU_lift_Cstop	T	1
TU_lift_run	P	TU_lift_csc	T	1
TU_out	P	table_unlock_output_area	T	1
TU_rot_rs	P	TU_rot_Pstop	T	1
TU_rot_run	P	TU_rot_Cstop	T	1
TU_rot_run	P	TU_rot_csc	T	1
arm1_backward	P	A1L_ret_Pstop	T	1
arm1_backward	P	A1L_ret_csc	T	1
arm1_backward	P	A1U_ret_Pstop	T	1
arm1_backward	P	A1U_ret_csc	T	1
arm1_forward	P	A1L_ext_Pstop	T	1
arm1_forward	P	A1L_ext_csc	T	1
arm1_forward	P	A1U_ext_Pstop	T	1
arm1_forward	P	A1U_ext_csc	T	1
arm1_having_swivel_1	P	arm1_unlock_swivel_1	T	1
arm1_having_swivel_2	P	arm1_unlock_swivel_2	T	1
arm1_magnet_off	P	A1L_grasp	T	1
arm1_magnet_on	P	A1U_ungrasp	T	1
arm1_pick_up_angle	P	A1L_rot1_Cstop	T	1
arm1_pick_up_angle	P	A1L_rot2_Cstop	T	1
arm1_pick_up_angle	P	A1L_rot3_Cstop	T	1
arm1_pick_up_angle	P	A1U_c1	T	1
arm1_pick_up_angle	P	A1U_rot1_csc	T	1
arm1_pick_up_angle	P	A2L_c1	T	1
arm1_pick_up_angle	P	A2L_rot1_csc	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
arm1_pick_up_angle	P	A2U_c1	T	1
arm1_pick_up_angle	P	A2U_rot1_csc	T	1
arm1_pick_up_ext	P	A1L_ext_Cstop	T	1
arm1_pick_up_ext	P	A1L_ret_csc	T	1
arm1_release_angle	P	A1L_c1	T	1
arm1_release_angle	P	A1L_rot1_csc	T	1
arm1_release_angle	P	A1U_rot1_Cstop	T	1
arm1_release_angle	P	A1U_rot2_Cstop	T	1
arm1_release_angle	P	A1U_rot3_Cstop	T	1
arm1_release_angle	P	A2L_c2	T	1
arm1_release_angle	P	A2L_rot2_csc	T	1
arm1_release_angle	P	A2U_c2	T	1
arm1_release_angle	P	A2U_rot2_csc	T	1
arm1_release_ext	P	A1U_ext_Cstop	T	1
arm1_release_ext	P	A1U_ret_csc	T	1
arm1_retract_ext	P	A1L_ext_csc	T	1
arm1_retract_ext	P	A1L_ret_Cstop	T	1
arm1_retract_ext	P	A1U_ext_csc	T	1
arm1_retract_ext	P	A1U_ret_Cstop	T	1
arm1_stop	P	A1L_ext_Pstart	T	1
arm1_stop	P	A1L_ret_Pstart	T	1
arm1_stop	P	A1U_ext_Pstart	T	1
arm1_stop	P	A1U_ret_Pstart	T	1
arm1_store_free	P	arm1_lock_input_area	T	1
arm1_storing	P	arm1_lock_output_area	T	1
arm1_waiting_for_-swivel_1	P	arm1_lock_swivel_1	T	1
arm1_waiting_for_-swivel_2	P	arm1_lock_swivel_2	T	1
arm2_backward	P	A2L_ret_Pstop	T	1
arm2_backward	P	A2L_ret_csc	T	1
arm2_backward	P	A2U_ret_Pstop	T	1
arm2_backward	P	A2U_ret_csc	T	1
arm2_forward	P	A2L_ext_Pstop	T	1
arm2_forward	P	A2L_ext_csc	T	1
arm2_forward	P	A2U_ext_Pstop	T	1
arm2_forward	P	A2U_ext_csc	T	1
arm2_having_swivel_1	P	arm2_unlock_swivel_1	T	1
arm2_having_swivel_2	P	arm2_unlock_swivel_2	T	1
arm2_magnet_off	P	A2L_grasp	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
arm2_magnet_on	P	A2U_ungrasp	T	1
arm2_pick_up_angle	P	A1L_c2	T	1
arm2_pick_up_angle	P	A1L_rot2_csc	T	1
arm2_pick_up_angle	P	A1U_c2	T	1
arm2_pick_up_angle	P	A1U_rot2_csc	T	1
arm2_pick_up_angle	P	A2L_rot1_Cstop	T	1
arm2_pick_up_angle	P	A2L_rot2_Cstop	T	1
arm2_pick_up_angle	P	A2L_rot3_Cstop	T	1
arm2_pick_up_angle	P	A2U_c3	T	1
arm2_pick_up_angle	P	A2U_rot3_csc	T	1
arm2_pick_up_ext	P	A2L_ext_Cstop	T	1
arm2_pick_up_ext	P	A2L_ret_csc	T	1
arm2_release_angle	P	A1L_c3	T	1
arm2_release_angle	P	A1L_rot3_csc	T	1
arm2_release_angle	P	A1U_c3	T	1
arm2_release_angle	P	A1U_rot3_csc	T	1
arm2_release_angle	P	A2L_c3	T	1
arm2_release_angle	P	A2L_rot3_csc	T	1
arm2_release_angle	P	A2U_rot1_Cstop	T	1
arm2_release_angle	P	A2U_rot2_Cstop	T	1
arm2_release_angle	P	A2U_rot3_Cstop	T	1
arm2_release_ext	P	A2U_ext_Cstop	T	1
arm2_release_ext	P	A2U_ret_csc	T	1
arm2_retract_ext	P	A2L_ext_csc	T	1
arm2_retract_ext	P	A2L_ret_Cstop	T	1
arm2_retract_ext	P	A2U_ext_csc	T	1
arm2_retract_ext	P	A2U_ret_Cstop	T	1
arm2_stop	P	A2L_ext_Pstart	T	1
arm2_stop	P	A2L_ret_Pstart	T	1
arm2_stop	P	A2U_ext_Pstart	T	1
arm2_stop	P	A2U_ret_Pstart	T	1
arm2_store_free	P	arm2_lock_input_area	T	1
arm2_storing	P	arm2_lock_output_area	T	1
arm2_waiting_for_-swivel_1	P	arm2_lock_swivel_1	T	1
arm2_waiting_for_-swivel_2	P	arm2_lock_swivel_2	T	1
belt1_light_barrier_false	P	FB_deliver_Cstop	T	1
belt1_light_barrier_false	P	FB_trans_csc	T	1
belt1_light_barrier_true	P	FB_deliver_csc	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
belt1_light_barrier_true	P	FB_trans_Cstop	T	1
belt1_start	P	FB_deliver_Pstop	T	1
belt1_start	P	FB_deliver_csc	T	1
belt1_start	P	FB_trans_Pstop	T	1
belt1_start	P	FB_trans_csc	T	1
belt1_stop	P	FB_deliver_Pstart	T	1
belt1_stop	P	FB_trans_Pstart	T	1
belt2_light_barrier_false	P	DB_deliver_Cstop	T	1
belt2_light_barrier_false	P	DB_trans_csc	T	1
belt2_light_barrier_true	P	DB_deliver_csc	T	1
belt2_light_barrier_true	P	DB_trans_Cstop	T	1
belt2_start	P	DB_deliver_Pstop	T	1
belt2_start	P	DB_deliver_csc	T	1
belt2_start	P	DB_trans_Pstop	T	1
belt2_start	P	DB_trans_csc	T	1
belt2_stop	P	DB_deliver_Pstart	T	1
belt2_stop	P	DB_trans_Pstart	T	1
blank_forged	P	PU_lower_Pstart	T	1
ch_A1P_free	P	arm1_lock_output_area	T	1
ch_A1P_full	P	press_lock_input_area	T	1
ch_A2D_free	P	arm2_lock_output_area	T	1
ch_A2D_full	P	deposit_belt_lock_input_-area	T	1
ch_CF_free	P	crane_lock_output_area	T	1
ch_CF_full	P	feed_belt_lock_input_area	T	1
ch_DC_free	P	deposit_belt_lock_-output_area	T	1
ch_DC_full	P	crane_lock_input_area	T	1
ch_FT_free	P	feed_belt_lock_output_-area	T	1
ch_FT_full	P	table_lock_input_area	T	1
ch_PA2_free	P	press_lock_output_area	T	1
ch_PA2_full	P	arm2_lock_input_area	T	1
ch_TA1_free	P	table_lock_output_area	T	1
ch_TA1_full	P	arm1_lock_input_area	T	1
crane_above_deposit_belt	P	CL_trans_csc	T	1
crane_above_deposit_belt	P	CU_trans_Cstop	T	1
crane_above_feed_belt	P	CL_trans_Cstop	T	1
crane_above_feed_belt	P	CU_trans_csc	T	1
crane_lift	P	CL_lift_Pstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
crane_lift	P	CL_lift_csc	T	1
crane_lift	P	CU_lift_Pstop	T	1
crane_lift	P	CU_lift_csc	T	1
crane_lower	P	CL_lower_Pstop	T	1
crane_lower	P	CL_lower_csc	T	1
crane_lower	P	CU_lower_Pstop	T	1
crane_lower	P	CU_lower_csc	T	1
crane_mag_off	P	CL_grasp	T	1
crane_mag_on	P	CU_ungrasp	T	1
crane_pick_up_height	P	CL_lift_csc	T	1
crane_pick_up_height	P	CL_lower_Cstop	T	1
crane_release_height	P	CU_lift_csc	T	1
crane_release_height	P	CU_lower_Cstop	T	1
crane_stop_h	P	CL_trans_Pstart	T	1
crane_stop_h	P	CU_trans_Pstart	T	1
crane_stop_v	P	CL_lift_Pstart	T	1
crane_stop_v	P	CL_lower_Pstart	T	1
crane_stop_v	P	CU_lift_Pstart	T	1
crane_stop_v	P	CU_lower_Pstart	T	1
crane_store_free	P	crane_lock_input_area	T	1
crane_storing	P	crane_lock_output_area	T	1
crane_to_belt1	P	CL_trans_Pstop	T	1
crane_to_belt1	P	CL_trans_csc	T	1
crane_to_belt2	P	CU_trans_Pstop	T	1
crane_to_belt2	P	CU_trans_csc	T	1
crane_transport_height	P	CL_lift_Cstop	T	1
crane_transport_height	P	CL_lower_csc	T	1
crane_transport_height	P	CU_lift_Cstop	T	1
crane_transport_height	P	CU_lower_csc	T	1
deposit_belt_empty	P	deposit_belt_unlock_-output_area	T	1
deposit_belt_idle	P	deposit_belt_lock_input_-area	T	1
deposit_belt_occupied	P	deposit_belt_lock_-output_area	T	1
feed_belt_empty	P	feed_belt_unlock_-output_area	T	1
feed_belt_idle	P	feed_belt_lock_input_area	T	1
feed_belt_occupied	P	feed_belt_lock_output_-area	T	1
forge_rs	P	forge_Pstop	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
forge_run	P	forge_Cstop	T	1
forge_run	P	forge_csc	T	1
press_at_lower_pos	P	PL_lower_csc	T	1
press_at_lower_pos	P	PU_lower_Cstop	T	1
press_at_middle_pos	P	PL_lower_Cstop	T	1
press_at_middle_pos	P	forge_csc	T	1
press_at_upper_pos	P	PU_lower_csc	T	1
press_at_upper_pos	P	forge_Cstop	T	1
press_down	P	PU_lower_Pstop	T	1
press_down	P	PU_lower_csc	T	1
press_ready_for_loading	P	press_lock_input_area	T	1
press_ready_for_-unloading	P	press_lock_output_area	T	1
press_stop	P	PL_lower_Pstart	T	1
press_stop	P	PU_lower_Pstart	T	1
press_stop	P	forge_Pstart	T	1
press_up	P	PL_lower_Pstop	T	1
press_up	P	PL_lower_csc	T	1
press_upward	P	forge_Pstop	T	1
press_upward	P	forge_csc	T	1
robot_left	P	A1L_rot1_Pstop	T	1
robot_left	P	A1L_rot1_csc	T	1
robot_left	P	A1L_rot2_Pstop	T	1
robot_left	P	A1L_rot2_csc	T	1
robot_left	P	A1L_rot3_Pstop	T	1
robot_left	P	A1L_rot3_csc	T	1
robot_left	P	A1U_rot3_Pstop	T	1
robot_left	P	A1U_rot3_csc	T	1
robot_left	P	A2L_rot1_Pstop	T	1
robot_left	P	A2L_rot1_csc	T	1
robot_left	P	A2L_rot2_Pstop	T	1
robot_left	P	A2L_rot2_csc	T	1
robot_left	P	A2L_rot3_Pstop	T	1
robot_left	P	A2L_rot3_csc	T	1
robot_left	P	A2U_rot2_Pstop	T	1
robot_left	P	A2U_rot2_csc	T	1
robot_right	P	A1U_rot1_Pstop	T	1
robot_right	P	A1U_rot1_csc	T	1
robot_right	P	A1U_rot2_Pstop	T	1
robot_right	P	A1U_rot2_csc	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
robot_right	P	A2U_rot1_Pstop	T	1
robot_right	P	A2U_rot1_csc	T	1
robot_right	P	A2U_rot3_Pstop	T	1
robot_right	P	A2U_rot3_csc	T	1
robot_stop	P	A1L_rot1_Pstart	T	1
robot_stop	P	A1L_rot2_Pstart	T	1
robot_stop	P	A1L_rot3_Pstart	T	1
robot_stop	P	A1U_rot1_Pstart	T	1
robot_stop	P	A1U_rot2_Pstart	T	1
robot_stop	P	A1U_rot3_Pstart	T	1
robot_stop	P	A2L_rot1_Pstart	T	1
robot_stop	P	A2L_rot2_Pstart	T	1
robot_stop	P	A2L_rot3_Pstart	T	1
robot_stop	P	A2U_rot1_Pstart	T	1
robot_stop	P	A2U_rot2_Pstart	T	1
robot_stop	P	A2U_rot3_Pstart	T	1
swivel	P	arm1_lock_swivel_1	T	1
swivel	P	arm1_lock_swivel_2	T	1
swivel	P	arm2_lock_swivel_1	T	1
swivel	P	arm2_lock_swivel_2	T	1
table_at_load_angle	P	TL_lower_Pstart	T	1
table_at_unload_angle	P	TU_lift_Pstart	T	1
table_bottom_pos	P	TL_lower_Cstop	T	1
table_bottom_pos	P	TU_lift_csc	T	1
table_load_angle	P	TL_rot_Cstop	T	1
table_load_angle	P	TU_rot_csc	T	1
table_ready_for_loading	P	table_lock_input_area	T	1
table_ready_for_-unloading	P	table_lock_output_area	T	1
table_right	P	TL_rot_Pstop	T	1
table_right	P	TL_rot_csc	T	1
table_right	P	TU_rot_Pstop	T	1
table_right	P	TU_rot_csc	T	1
table_stop_h	P	TL_rot_Pstart	T	1
table_stop_h	P	TU_rot_Pstart	T	1
table_stop_v	P	TL_lower_Pstart	T	1
table_stop_v	P	TU_lift_Pstart	T	1
table_top_pos	P	TL_lower_csc	T	1
table_top_pos	P	TU_lift_Cstop	T	1
table_unload_angle	P	TL_rot_csc	T	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
table_unload_angle	P	TU_rot_Cstop	T	1
table_upward	P	TL_lower_Pstop	T	1
table_upward	P	TL_lower_csc	T	1
table_upward	P	TU_lift_Pstop	T	1
table_upward	P	TU_lift_csc	T	1
A1L_c1	T	A1L_rot1_in	P	1
A1L_c1	T	arm1_release_angle	P	1
A1L_c2	T	A1L_rot2_in	P	1
A1L_c2	T	arm2_pick_up_angle	P	1
A1L_c3	T	A1L_rot3_in	P	1
A1L_c3	T	arm2_release_angle	P	1
A1L_ext_Cstop	T	A1L_ext_rs	P	1
A1L_ext_Cstop	T	arm1_pick_up_ext	P	1
A1L_ext_Pstart	T	A1L_ext_run	P	1
A1L_ext_Pstart	T	arm1_forward	P	1
A1L_ext_Pstop	T	A1_extended	P	1
A1L_ext_Pstop	T	arm1_stop	P	1
A1L_ext_csc	T	A1L_ext_run	P	1
A1L_ext_csc	T	arm1_forward	P	1
A1L_ext_csc	T	arm1_pick_up_ext	P	1
A1L_grasp	T	A1L_loaded	P	1
A1L_grasp	T	arm1_magnet_on	P	1
A1L_ret_Cstop	T	A1L_ret_rs	P	1
A1L_ret_Cstop	T	arm1_retract_ext	P	1
A1L_ret_Pstart	T	A1L_ret_run	P	1
A1L_ret_Pstart	T	arm1_backward	P	1
A1L_ret_Pstop	T	A1L_out	P	1
A1L_ret_Pstop	T	arm1_stop	P	1
A1L_ret_csc	T	A1L_ret_run	P	1
A1L_ret_csc	T	arm1_backward	P	1
A1L_ret_csc	T	arm1_retract_ext	P	1
A1L_rot1_Cstop	T	A1L_rot1_rs	P	1
A1L_rot1_Cstop	T	arm1_pick_up_angle	P	1
A1L_rot1_Pstart	T	A1L_rot1_run	P	1
A1L_rot1_Pstart	T	robot_left	P	1
A1L_rot1_Pstop	T	A1L_rotated	P	1
A1L_rot1_Pstop	T	robot_stop	P	1
A1L_rot1_csc	T	A1L_rot1_run	P	1
A1L_rot1_csc	T	arm1_pick_up_angle	P	1
A1L_rot1_csc	T	robot_left	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A1L_rot2_Cstop	T	A1L_rot2_rs	P	1
A1L_rot2_Cstop	T	arm1_pick_up_angle	P	1
A1L_rot2_Pstart	T	A1L_rot2_run	P	1
A1L_rot2_Pstart	T	robot_left	P	1
A1L_rot2_Pstop	T	A1L_rotated	P	1
A1L_rot2_Pstop	T	robot_stop	P	1
A1L_rot2_csc	T	A1L_rot2_run	P	1
A1L_rot2_csc	T	arm1_pick_up_angle	P	1
A1L_rot2_csc	T	robot_left	P	1
A1L_rot3_Cstop	T	A1L_rot3_rs	P	1
A1L_rot3_Cstop	T	arm1_pick_up_angle	P	1
A1L_rot3_Pstart	T	A1L_rot3_run	P	1
A1L_rot3_Pstart	T	robot_left	P	1
A1L_rot3_Pstop	T	A1L_rotated	P	1
A1L_rot3_Pstop	T	robot_stop	P	1
A1L_rot3_csc	T	A1L_rot3_run	P	1
A1L_rot3_csc	T	arm1_pick_up_angle	P	1
A1L_rot3_csc	T	robot_left	P	1
A1U_c1	T	A1U_rot1_in	P	1
A1U_c1	T	arm1_pick_up_angle	P	1
A1U_c2	T	A1U_rot2_in	P	1
A1U_c2	T	arm2_pick_up_angle	P	1
A1U_c3	T	A1U_rot3_in	P	1
A1U_c3	T	arm2_release_angle	P	1
A1U_ext_Cstop	T	A1U_ext_rs	P	1
A1U_ext_Cstop	T	arm1_release_ext	P	1
A1U_ext_Pstart	T	A1U_ext_run	P	1
A1U_ext_Pstart	T	arm1_forward	P	1
A1U_ext_Pstop	T	A1U_extendet	P	1
A1U_ext_Pstop	T	arm1_stop	P	1
A1U_ext_csc	T	A1U_ext_run	P	1
A1U_ext_csc	T	arm1_forward	P	1
A1U_ext_csc	T	arm1_release_ext	P	1
A1U_ret_Cstop	T	A1U_ret_rs	P	1
A1U_ret_Cstop	T	arm1_retract_ext	P	1
A1U_ret_Pstart	T	A1U_ret_run	P	1
A1U_ret_Pstart	T	arm1_backward	P	1
A1U_ret_Pstop	T	A1U_out	P	1
A1U_ret_Pstop	T	arm1_stop	P	1
A1U_ret_csc	T	A1U_ret_run	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A1U_ret_csc	T	arm1_backward	P	1
A1U_ret_csc	T	arm1_retract_ext	P	1
A1U_rot1_Cstop	T	A1U_rot1_rs	P	1
A1U_rot1_Cstop	T	arm1_release_angle	P	1
A1U_rot1_Pstart	T	A1U_rot1_run	P	1
A1U_rot1_Pstart	T	robot_right	P	1
A1U_rot1_Pstop	T	A1U_rotated	P	1
A1U_rot1_Pstop	T	robot_stop	P	1
A1U_rot1_csc	T	A1U_rot1_run	P	1
A1U_rot1_csc	T	arm1_release_angle	P	1
A1U_rot1_csc	T	robot_right	P	1
A1U_rot2_Cstop	T	A1U_rot2_rs	P	1
A1U_rot2_Cstop	T	arm1_release_angle	P	1
A1U_rot2_Pstart	T	A1U_rot2_run	P	1
A1U_rot2_Pstart	T	robot_right	P	1
A1U_rot2_Pstop	T	A1U_rotated	P	1
A1U_rot2_Pstop	T	robot_stop	P	1
A1U_rot2_csc	T	A1U_rot2_run	P	1
A1U_rot2_csc	T	arm1_release_angle	P	1
A1U_rot2_csc	T	robot_right	P	1
A1U_rot3_Cstop	T	A1U_rot3_rs	P	1
A1U_rot3_Cstop	T	arm1_release_angle	P	1
A1U_rot3_Pstart	T	A1U_rot3_run	P	1
A1U_rot3_Pstart	T	robot_left	P	1
A1U_rot3_Pstop	T	A1U_rotated	P	1
A1U_rot3_Pstop	T	robot_stop	P	1
A1U_rot3_csc	T	A1U_rot3_run	P	1
A1U_rot3_csc	T	arm1_release_angle	P	1
A1U_rot3_csc	T	robot_left	P	1
A1U_ungrasp	T	A1U_unloadet	P	1
A1U_ungrasp	T	arm1_magnet_off	P	1
A2L_c1	T	A2L_rot1_in	P	1
A2L_c1	T	arm1_pick_up_angle	P	1
A2L_c2	T	A2L_rot2_in	P	1
A2L_c2	T	arm1_release_angle	P	1
A2L_c3	T	A2L_rot3_in	P	1
A2L_c3	T	arm2_release_angle	P	1
A2L_ext_Cstop	T	A2L_ext_rs	P	1
A2L_ext_Cstop	T	arm2_pick_up_ext	P	1
A2L_ext_Pstart	T	A2L_ext_run	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A2L_ext_Pstart	T	arm2_forward	P	1
A2L_ext_Pstop	T	A2L_extended	P	1
A2L_ext_Pstop	T	arm2_stop	P	1
A2L_ext_csc	T	A2L_ext_run	P	1
A2L_ext_csc	T	arm2_forward	P	1
A2L_ext_csc	T	arm2_pick_up_ext	P	1
A2L_grasp	T	A2L_loaded	P	1
A2L_grasp	T	arm2_magnet_on	P	1
A2L_ret_Cstop	T	A2L_ret_rs	P	1
A2L_ret_Cstop	T	arm2_retract_ext	P	1
A2L_ret_Pstart	T	A2L_ret_run	P	1
A2L_ret_Pstart	T	arm2_backward	P	1
A2L_ret_Pstop	T	A2L_out	P	1
A2L_ret_Pstop	T	arm2_stop	P	1
A2L_ret_csc	T	A2L_ret_run	P	1
A2L_ret_csc	T	arm2_backward	P	1
A2L_ret_csc	T	arm2_retract_ext	P	1
A2L_rot1_Cstop	T	A2L_rot1_rs	P	1
A2L_rot1_Cstop	T	arm2_pick_up_angle	P	1
A2L_rot1_Pstart	T	A2L_rot1_run	P	1
A2L_rot1_Pstart	T	robot_left	P	1
A2L_rot1_Pstop	T	A2L_rotated	P	1
A2L_rot1_Pstop	T	robot_stop	P	1
A2L_rot1_csc	T	A2L_rot1_run	P	1
A2L_rot1_csc	T	arm2_pick_up_angle	P	1
A2L_rot1_csc	T	robot_left	P	1
A2L_rot2_Cstop	T	A2L_rot2_rs	P	1
A2L_rot2_Cstop	T	arm2_pick_up_angle	P	1
A2L_rot2_Pstart	T	A2L_rot2_run	P	1
A2L_rot2_Pstart	T	robot_left	P	1
A2L_rot2_Pstop	T	A2L_rotated	P	1
A2L_rot2_Pstop	T	robot_stop	P	1
A2L_rot2_csc	T	A2L_rot2_run	P	1
A2L_rot2_csc	T	arm2_pick_up_angle	P	1
A2L_rot2_csc	T	robot_left	P	1
A2L_rot3_Cstop	T	A2L_rot3_rs	P	1
A2L_rot3_Cstop	T	arm2_pick_up_angle	P	1
A2L_rot3_Pstart	T	A2L_rot3_run	P	1
A2L_rot3_Pstart	T	robot_left	P	1
A2L_rot3_Pstop	T	A2L_rotated	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A2L_rot3_Pstop	T	robot_stop	P	1
A2L_rot3_csc	T	A2L_rot3_run	P	1
A2L_rot3_csc	T	arm2_pick_up_angle	P	1
A2L_rot3_csc	T	robot_left	P	1
A2U_c1	T	A2U_rot1_in	P	1
A2U_c1	T	arm1_pick_up_angle	P	1
A2U_c2	T	A2U_rot2_in	P	1
A2U_c2	T	arm1_release_angle	P	1
A2U_c3	T	A2U_rot3_in	P	1
A2U_c3	T	arm2_pick_up_angle	P	1
A2U_ext_Cstop	T	A2U_ext_rs	P	1
A2U_ext_Cstop	T	arm2_release_ext	P	1
A2U_ext_Pstart	T	A2U_ext_run	P	1
A2U_ext_Pstart	T	arm2_forward	P	1
A2U_ext_Pstop	T	A2U_extended	P	1
A2U_ext_Pstop	T	arm2_stop	P	1
A2U_ext_csc	T	A2U_ext_run	P	1
A2U_ext_csc	T	arm2_forward	P	1
A2U_ext_csc	T	arm2_release_ext	P	1
A2U_ret_Cstop	T	A2U_ret_rs	P	1
A2U_ret_Cstop	T	arm2_retract_ext	P	1
A2U_ret_Pstart	T	A2U_ret_run	P	1
A2U_ret_Pstart	T	arm2_backward	P	1
A2U_ret_Pstop	T	A2U_out	P	1
A2U_ret_Pstop	T	arm2_stop	P	1
A2U_ret_csc	T	A2U_ret_run	P	1
A2U_ret_csc	T	arm2_backward	P	1
A2U_ret_csc	T	arm2_retract_ext	P	1
A2U_rot1_Cstop	T	A2U_rot1_rs	P	1
A2U_rot1_Cstop	T	arm2_release_angle	P	1
A2U_rot1_Pstart	T	A2U_rot1_run	P	1
A2U_rot1_Pstart	T	robot_right	P	1
A2U_rot1_Pstop	T	A2U_rotated	P	1
A2U_rot1_Pstop	T	robot_stop	P	1
A2U_rot1_csc	T	A2U_rot1_run	P	1
A2U_rot1_csc	T	arm2_release_angle	P	1
A2U_rot1_csc	T	robot_right	P	1
A2U_rot2_Cstop	T	A2U_rot2_rs	P	1
A2U_rot2_Cstop	T	arm2_release_angle	P	1
A2U_rot2_Pstart	T	A2U_rot2_run	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A2U_rot2_Pstart	T	robot_left	P	1
A2U_rot2_Pstop	T	A2U_rotated	P	1
A2U_rot2_Pstop	T	robot_stop	P	1
A2U_rot2_csc	T	A2U_rot2_run	P	1
A2U_rot2_csc	T	arm2_release_angle	P	1
A2U_rot2_csc	T	robot_left	P	1
A2U_rot3_Cstop	T	A2U_rot3_rs	P	1
A2U_rot3_Cstop	T	arm2_release_angle	P	1
A2U_rot3_Pstart	T	A2U_rot3_run	P	1
A2U_rot3_Pstart	T	robot_right	P	1
A2U_rot3_Pstop	T	A2U_rotated	P	1
A2U_rot3_Pstop	T	robot_stop	P	1
A2U_rot3_csc	T	A2U_rot3_run	P	1
A2U_rot3_csc	T	arm2_release_angle	P	1
A2U_rot3_csc	T	robot_right	P	1
A2U_ungrasp	T	A2U_unloaded	P	1
A2U_ungrasp	T	arm2_magnet_off	P	1
CL_grasp	T	CL_loaded	P	1
CL_grasp	T	crane_mag_on	P	1
CL_lift_Cstop	T	CL_lift_rs	P	1
CL_lift_Cstop	T	crane_transport_height	P	1
CL_lift_Pstart	T	CL_lift_run	P	1
CL_lift_Pstart	T	crane_lift	P	1
CL_lift_Pstop	T	CL_ready_to_transport	P	1
CL_lift_Pstop	T	crane_stop_v	P	1
CL_lift_csc	T	CL_lift_run	P	1
CL_lift_csc	T	crane_lift	P	1
CL_lift_csc	T	crane_transport_height	P	1
CL_lower_Cstop	T	CL_lower_rs	P	1
CL_lower_Cstop	T	crane_pick_up_height	P	1
CL_lower_Pstart	T	CL_lower_run	P	1
CL_lower_Pstart	T	crane_lower	P	1
CL_lower_Pstop	T	CL_ready_to_grasp	P	1
CL_lower_Pstop	T	crane_stop_v	P	1
CL_lower_csc	T	CL_lower_run	P	1
CL_lower_csc	T	crane_lower	P	1
CL_lower_csc	T	crane_pick_up_height	P	1
CL_trans_Cstop	T	CL_trans_rs	P	1
CL_trans_Cstop	T	crane_above_feed_belt	P	1
CL_trans_Pstart	T	CL_trans_run	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
CL_trans_Pstart	T	crane_to_belt1	P	1
CL_trans_Pstop	T	CL_out	P	1
CL_trans_Pstop	T	crane_stop_h	P	1
CL_trans_csc	T	CL_trans_run	P	1
CL_trans_csc	T	crane_above_feed_belt	P	1
CL_trans_csc	T	crane_to_belt1	P	1
CU_lift_Cstop	T	CU_lift_rs	P	1
CU_lift_Cstop	T	crane_transport_height	P	1
CU_lift_Pstart	T	CU_lift_run	P	1
CU_lift_Pstart	T	crane_lift	P	1
CU_lift_Pstop	T	CU_ready_to_transport	P	1
CU_lift_Pstop	T	crane_stop_v	P	1
CU_lift_csc	T	CU_lift_run	P	1
CU_lift_csc	T	crane_lift	P	1
CU_lift_csc	T	crane_transport_height	P	1
CU_lower_Cstop	T	CU_lower_rs	P	1
CU_lower_Cstop	T	crane_release_height	P	1
CU_lower_Pstart	T	CU_lower_run	P	1
CU_lower_Pstart	T	crane_lower	P	1
CU_lower_Pstop	T	CU_ready_to_ungrasp	P	1
CU_lower_Pstop	T	crane_stop_v	P	1
CU_lower_csc	T	CU_lower_run	P	1
CU_lower_csc	T	crane_lower	P	1
CU_lower_csc	T	crane_release_height	P	1
CU_trans_Cstop	T	CU_trans_rs	P	1
CU_trans_Cstop	T	crane_above_deposit_belt	P	1
CU_trans_Pstart	T	CU_trans_run	P	1
CU_trans_Pstart	T	crane_to_belt2	P	1
CU_trans_Pstop	T	CU_out	P	1
CU_trans_Pstop	T	crane_stop_h	P	1
CU_trans_csc	T	CU_trans_run	P	1
CU_trans_csc	T	crane_above_deposit_belt	P	1
CU_trans_csc	T	crane_to_belt2	P	1
CU_ungrasp	T	CU_unloaded	P	1
CU_ungrasp	T	crane_mag_off	P	1
DB_deliver_Cstop	T	DB_deliver_rs	P	1
DB_deliver_Cstop	T	belt2_light_barrier_false	P	1
DB_deliver_Pstart	T	DB_deliver_run	P	1
DB_deliver_Pstart	T	belt2_start	P	1
DB_deliver_Pstop	T	DB_out	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
DB_deliver_Pstop	T	belt2_stop	P	1
DB_deliver_csc	T	DB_deliver_run	P	1
DB_deliver_csc	T	belt2_light_barrier_false	P	1
DB_deliver_csc	T	belt2_start	P	1
DB_trans_Cstop	T	DB_trans_rs	P	1
DB_trans_Cstop	T	belt2_light_barrier_true	P	1
DB_trans_Pstart	T	DB_trans_run	P	1
DB_trans_Pstart	T	belt2_start	P	1
DB_trans_Pstop	T	DB_at_end	P	1
DB_trans_Pstop	T	belt2_stop	P	1
DB_trans_csc	T	DB_trans_run	P	1
DB_trans_csc	T	belt2_light_barrier_true	P	1
DB_trans_csc	T	belt2_start	P	1
FB_deliver_Cstop	T	FB_deliver_rs	P	1
FB_deliver_Cstop	T	belt1_light_barrier_false	P	1
FB_deliver_Pstart	T	FB_deliver_run	P	1
FB_deliver_Pstart	T	belt1_start	P	1
FB_deliver_Pstop	T	FB_out	P	1
FB_deliver_Pstop	T	belt1_stop	P	1
FB_deliver_csc	T	FB_deliver_run	P	1
FB_deliver_csc	T	belt1_light_barrier_false	P	1
FB_deliver_csc	T	belt1_start	P	1
FB_trans_Cstop	T	FB_trans_rs	P	1
FB_trans_Cstop	T	belt1_light_barrier_true	P	1
FB_trans_Pstart	T	FB_trans_run	P	1
FB_trans_Pstart	T	belt1_start	P	1
FB_trans_Pstop	T	FB_at_end	P	1
FB_trans_Pstop	T	belt1_stop	P	1
FB_trans_csc	T	FB_trans_run	P	1
FB_trans_csc	T	belt1_light_barrier_true	P	1
FB_trans_csc	T	belt1_start	P	1
PL_lower_Cstop	T	PL_lower_rs	P	1
PL_lower_Cstop	T	press_at_middle_pos	P	1
PL_lower_Pstart	T	PL_lower_run	P	1
PL_lower_Pstart	T	press_up	P	1
PL_lower_Pstop	T	PL_out	P	1
PL_lower_Pstop	T	press_stop	P	1
PL_lower_csc	T	PL_lower_run	P	1
PL_lower_csc	T	press_at_middle_pos	P	1
PL_lower_csc	T	press_up	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
PU_lower_Cstop	T	PU_lower_rs	P	1
PU_lower_Cstop	T	press_at_lower_pos	P	1
PU_lower_Pstart	T	PU_lower_run	P	1
PU_lower_Pstart	T	press_down	P	1
PU_lower_Pstop	T	PU_out	P	1
PU_lower_Pstop	T	press_stop	P	1
PU_lower_csc	T	PU_lower_run	P	1
PU_lower_csc	T	press_at_lower_pos	P	1
PU_lower_csc	T	press_down	P	1
TL_lower_Cstop	T	TL_lower_rs	P	1
TL_lower_Cstop	T	table_bottom_pos	P	1
TL_lower_Pstart	T	TL_lower_run	P	1
TL_lower_Pstart	T	table_upward	P	1
TL_lower_Pstop	T	TL_out	P	1
TL_lower_Pstop	T	table_stop_v	P	1
TL_lower_csc	T	TL_lower_run	P	1
TL_lower_csc	T	table_bottom_pos	P	1
TL_lower_csc	T	table_upward	P	1
TL_rot_Cstop	T	TL_rot_rs	P	1
TL_rot_Cstop	T	table_load_angle	P	1
TL_rot_Pstart	T	TL_rot_run	P	1
TL_rot_Pstart	T	table_right	P	1
TL_rot_Pstop	T	table_at_load_angle	P	1
TL_rot_Pstop	T	table_stop_h	P	1
TL_rot_csc	T	TL_rot_run	P	1
TL_rot_csc	T	table_load_angle	P	1
TL_rot_csc	T	table_right	P	1
TU_lift_Cstop	T	TU_lift_rs	P	1
TU_lift_Cstop	T	table_top_pos	P	1
TU_lift_Pstart	T	TU_lift_run	P	1
TU_lift_Pstart	T	table_upward	P	1
TU_lift_Pstop	T	TU_out	P	1
TU_lift_Pstop	T	table_stop_v	P	1
TU_lift_csc	T	TU_lift_run	P	1
TU_lift_csc	T	table_top_pos	P	1
TU_lift_csc	T	table_upward	P	1
TU_rot_Cstop	T	TU_rot_rs	P	1
TU_rot_Cstop	T	table_unload_angle	P	1
TU_rot_Pstart	T	TU_rot_run	P	1
TU_rot_Pstart	T	table_right	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
TU_rot_Pstop	T	table_at_unload_angle	P	1
TU_rot_Pstop	T	table_stop_h	P	1
TU_rot_csc	T	TU_rot_run	P	1
TU_rot_csc	T	table_right	P	1
TU_rot_csc	T	table_unload_angle	P	1
arm1_lock_input_area	T	arm1_waiting_for_swivel_1	P	1
arm1_lock_output_area	T	arm1_waiting_for_swivel_2	P	1
arm1_lock_swivel_1	T	A1L_in	P	1
arm1_lock_swivel_2	T	A1U_in	P	1
arm1_unlock_input_area	T	arm1_having_swivel_1	P	1
arm1_unlock_input_area	T	ch_TA1_free	P	1
arm1_unlock_output_area	T	arm1_having_swivel_2	P	1
arm1_unlock_output_area	T	ch_A1P_full	P	1
arm1_unlock_swivel_1	T	arm1_storing	P	1
arm1_unlock_swivel_1	T	swivel	P	1
arm1_unlock_swivel_2	T	arm1_store_free	P	1
arm1_unlock_swivel_2	T	swivel	P	1
arm2_lock_input_area	T	arm2_waiting_for_swivel_1	P	1
arm2_lock_output_area	T	arm2_waiting_for_swivel_2	P	1
arm2_lock_swivel_1	T	A2L_in	P	1
arm2_lock_swivel_2	T	A2U_in	P	1
arm2_unlock_input_area	T	arm2_having_swivel_1	P	1
arm2_unlock_input_area	T	ch_PA2_free	P	1
arm2_unlock_output_area	T	arm2_having_swivel_2	P	1
arm2_unlock_output_area	T	ch_A2D_full	P	1
arm2_unlock_swivel_1	T	arm2_storing	P	1
arm2_unlock_swivel_1	T	swivel	P	1
arm2_unlock_swivel_2	T	arm2_store_free	P	1
arm2_unlock_swivel_2	T	swivel	P	1
crane_lock_input_area	T	CL_in	P	1
crane_lock_output_area	T	CU_in	P	1
crane_unlock_input_area	T	ch_DC_free	P	1
crane_unlock_input_area	T	crane_storing	P	1
crane_unlock_output_area	T	ch_CF_full	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
crane_unlock_output_area	T	crane_store_free	P	1
deposit_belt_lock_input_area	T	deposit_belt_occupied	P	1
deposit_belt_lock_output_area	T	DB_in	P	1
deposit_belt_unlock_input_area	T	ch_DC_full	P	1
deposit_belt_unlock_input_area	T	deposit_belt_empty	P	1
deposit_belt_unlock_output_area	T	ch_A2D_free	P	1
deposit_belt_unlock_output_area	T	deposit_belt_idle	P	1
feed_belt_lock_input_area	T	feed_belt_occupied	P	1
feed_belt_lock_output_area	T	FB_in	P	1
feed_belt_unlock_input_area	T	ch_FT_full	P	1
feed_belt_unlock_input_area	T	feed_belt_empty	P	1
feed_belt_unlock_output_area	T	ch_CF_free	P	1
feed_belt_unlock_output_area	T	feed_belt_idle	P	1
forge_Cstop	T	forge_rs	P	1
forge_Cstop	T	press_at_upper_pos	P	1
forge_Pstart	T	forge_run	P	1
forge_Pstart	T	press_upward	P	1
forge_Pstop	T	blank_forged	P	1
forge_Pstop	T	press_stop	P	1
forge_csc	T	forge_run	P	1
forge_csc	T	press_at_upper_pos	P	1
forge_csc	T	press_upward	P	1
press_lock_input_area	T	PU_in	P	1
press_lock_output_area	T	PL_in	P	1
press_unlock_input_area	T	ch_A1P_free	P	1
press_unlock_input_area	T	press_ready_for_loading	P	1
press_unlock_output_area	T	ch_PA2_full	P	1
press_unlock_output_area	T	press_ready_for_unloading	P	1
table_lock_input_area	T	TU_in	P	1

Continued on next page

Table 5: Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
table_lock_output_area	T	TL_in	P	1
table_unlock_input_area	T	ch_FT_free	P	1
table_unlock_input_area	T	table_ready_for_loading	P	1
table_unlock_output_area	T	ch_TA1_full	P	1
table_unlock_output_area	T	table_ready_for_unloading	P	1

2.6 Comments

Table 6: Comments Table

NET REFERENCE	COMMENT
TopLevel	coarse structure of the full refined closed system
press(5)	table / press (mutual exclusive input / output)
press(5)	table / press (mutual exclusive input / output)
deposit_belt(7)	belts (dependent input / output)
deposit_belt(7)	belts (dependent input / output)
arm2(9)	arms (independent input/output)
arm2(9)	arms (independent input/output)
crane(12)	crane (independent input/output)

2.7 Images

No elements available for description

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

No elements available for description

3.2 Functions

No elements available for description

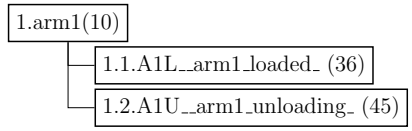
4 Hierarchy

This section contains information about the net hierarchy.

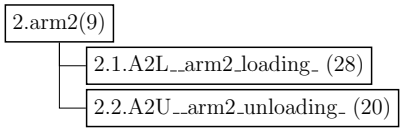
4.1 Hierarchy Tree

This section contains the hierarchical tree structure for selected levels.

1. arm1 (10)



2. arm2 (9)

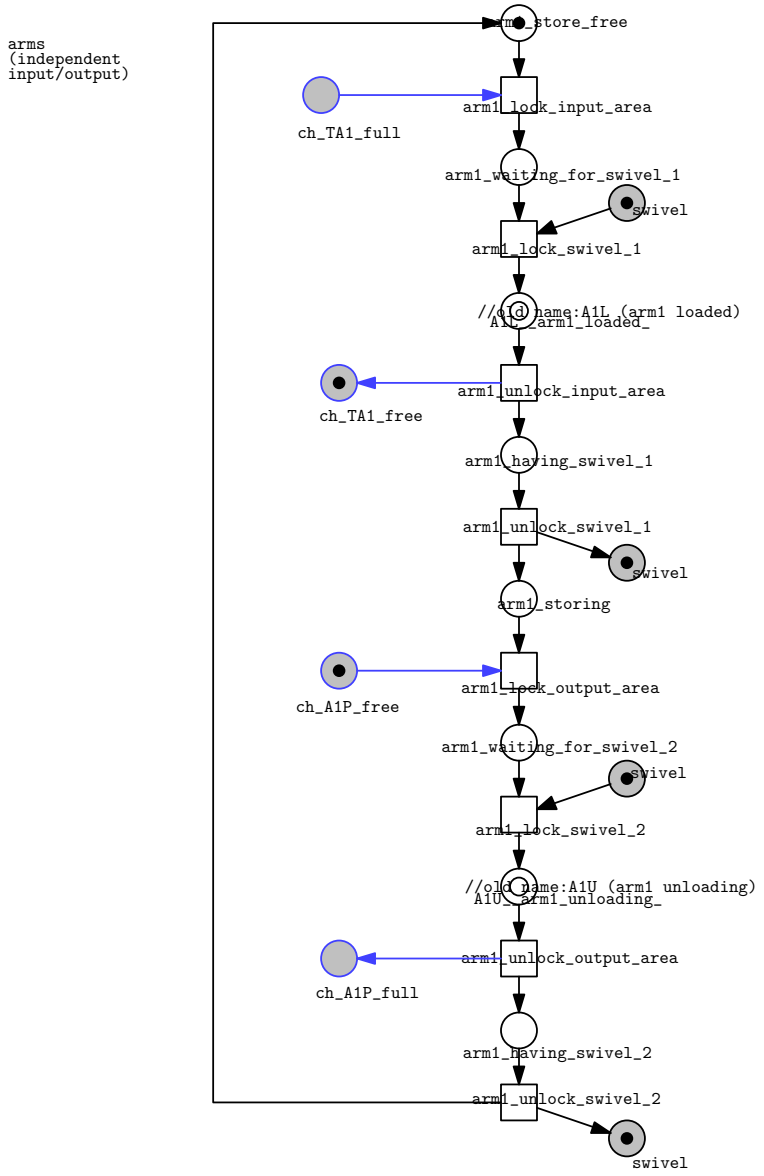


4. deposit_belt (7)

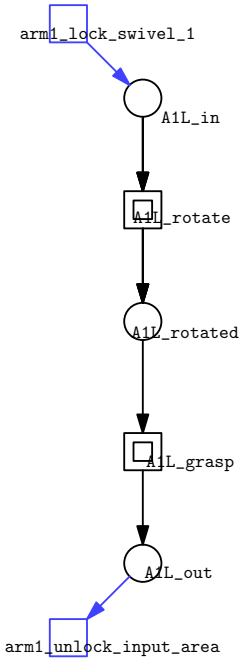
4.2 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

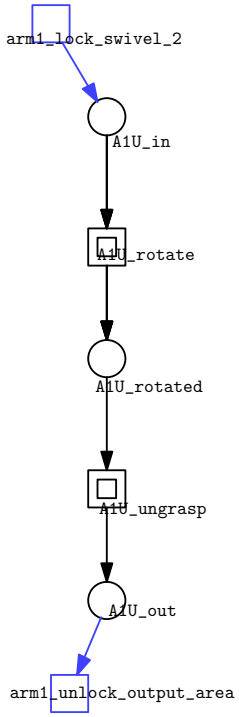
1. arm1 (10)



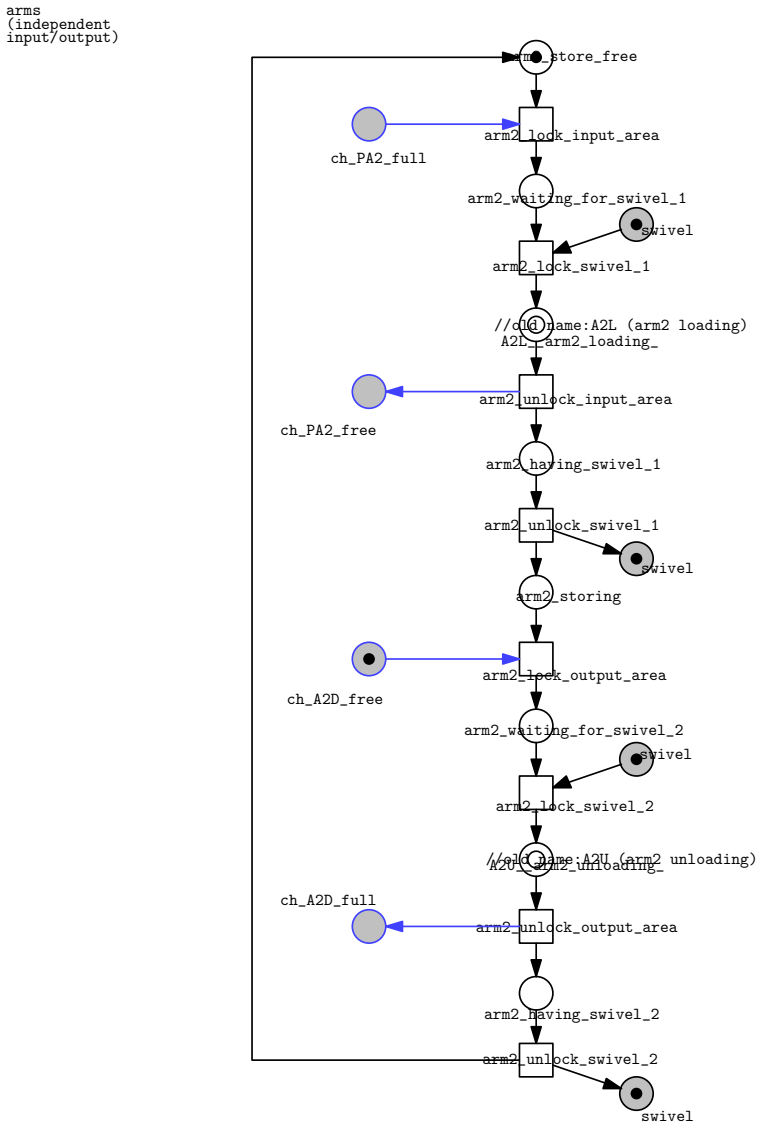
1.1. A1L__arm1_loaded_ (36)



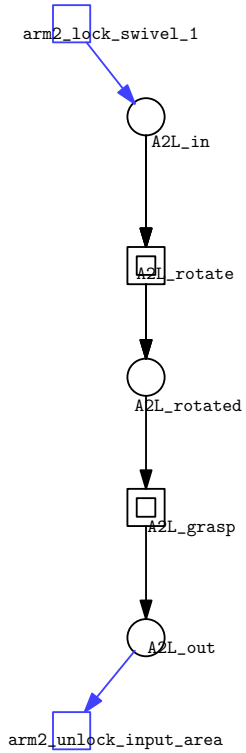
1.2. A1U__arm1_unloading_ (45)



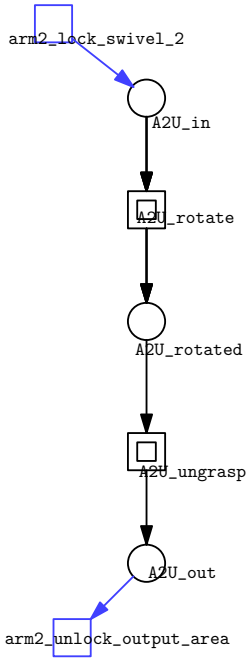
2. arm2 (9)



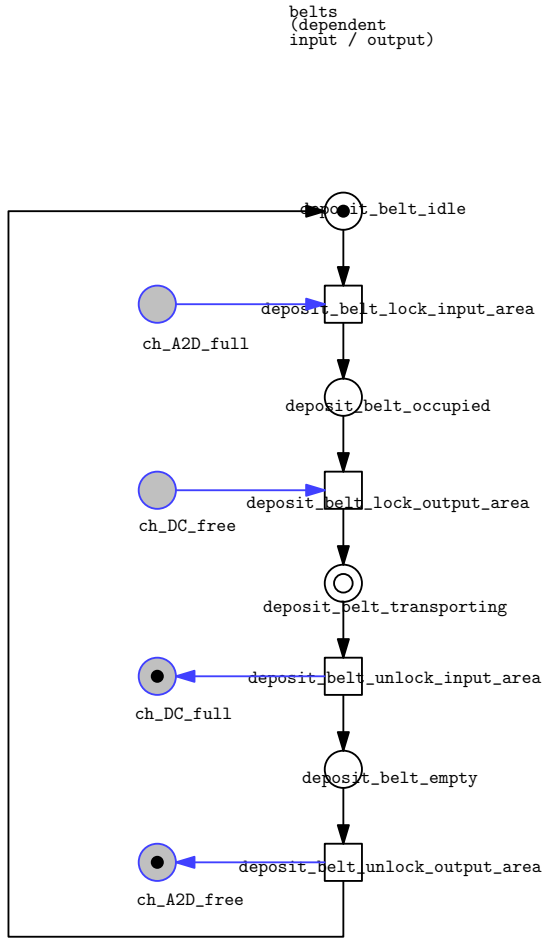
2.1. A2L__arm2_loading_ (28)



2.2. A2U__arm2_unloading_ (20)



4. deposit_belt (7)



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

[3] M Heiner and P Deussen: Petri Net Based Qualitative Analysis-A Case Study; Brandenburg Technical University of Cottbus, Department of Computer Science, Technical Report I-08/1995.

[4] M Heiner, P Deussen and J Spranger: A Case Study in Design and Verification of Manufacturing Systems with Hierarchical Petri Nets; International Journal of Advanced Manufacturing Technology 1999, volume 15, pages 139-152, 1999.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.1.2 Vilar.xpn (Extended Petri Net)

This is a single level net with a simple structure. The top level for this file is shown in Figure A.39. Customizations done in Snoopy2LaTeX dialog:

- **Places** selected against regex - "A" (Figure A.40)
- **Reset Arcs** and **Equal Arcs** deselected for export (Figure A.40)
- **Hierarchy** section reordered to precede 'Graph Elements'
- **Generate Hierarchy Tree** deselected for export

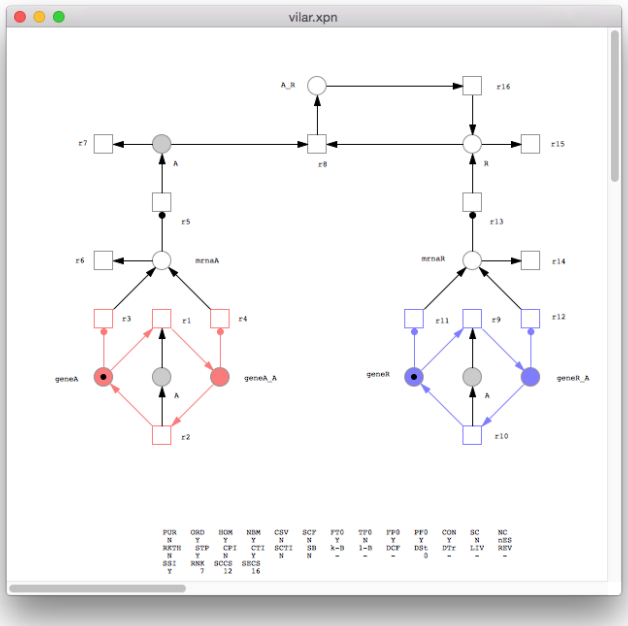


Figure A.39: Vilar - Snoopy file

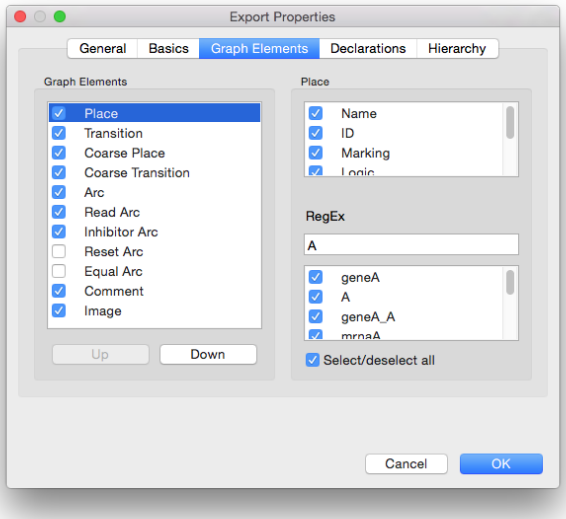
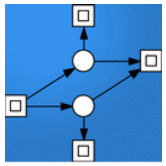


Figure A.40: Custom Graph Elements in Snoopy2LaTeX

The following pages show the generated PDF report.



vilar.xpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

vilar.xpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 14:49

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: vilar.xpn
File Path: /Users/anjali/Dropbox/Anjali-Sharma/examples/ExtendedPetriNet/villar/vilar.xpn
Creation Timestamp: 2014-07-12 20:51:07

1.2 Net Informations

Net Class: Extended Petri Net

Element	Count
Place	9
Transition	16
Arc	27
Read Arc	6
Inhibitor Arc	0
Reset Arc	0
Equal Arc	0

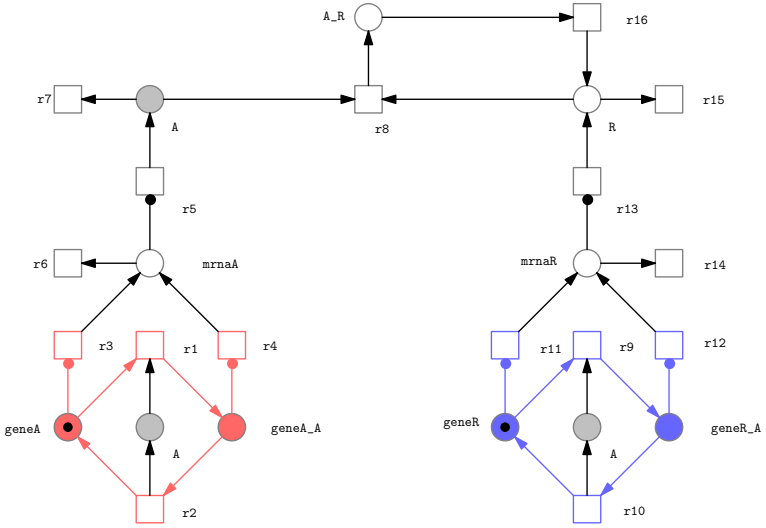
2 Hierarchy

This section contains information about the net hierarchy.

2.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level




```
PUR ORD HOM NBM CSV SCF FTO TFO FPO PFO CON SC NC
N Y Y Y N N Y N Y Y Y N nES
RKTH STP CPI CTI SCTI SB k-B 1-B DCF DSt DTr LIV REV
N Y N Y N N - - - 0 - - -
SSI RNK SCCS SECS
Y 7 12 16
```


3 Graph Elements

This section contains information related to graph elements specific to the net.

3.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
geneA	0	1	✗	TopLevel
A	1	0	✓	TopLevel
geneA_A	2	0	✗	TopLevel
mrnaA	3	0	✗	TopLevel
geneR_A	5	0	✗	TopLevel
A_R	8	0	✗	TopLevel

3.2 Transitions

Table 2: Transitions Table

NAME	ID	LOG.	CROSS REFS.
r1	0	✗	TopLevel
r2	1	✗	TopLevel
r3	2	✗	TopLevel
r4	3	✗	TopLevel
r6	4	✗	TopLevel
r5	5	✗	TopLevel
r7	6	✗	TopLevel
r15	7	✗	TopLevel
r13	8	✗	TopLevel
r14	9	✗	TopLevel
r12	10	✗	TopLevel
r11	11	✗	TopLevel
r10	12	✗	TopLevel
r9	13	✗	TopLevel
r8	14	✗	TopLevel
r16	15	✗	TopLevel

3.3 Coarse Places

No elements available for description

3.4 Coarse Transitions

No elements available for description

3.5 Arcs

Table 3: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
A	P	r1	T	1
A	P	r7	T	1
A	P	r8	T	1
A	P	r9	T	1
A_R	P	r16	T	1
R	P	r15	T	1
R	P	r8	T	1
geneA	P	r1	T	1
geneA_A	P	r2	T	1
geneR	P	r9	T	1
geneR_A	P	r10	T	1
mrnaA	P	r6	T	1
mrnaR	P	r14	T	1
r1	T	geneA_A	P	1
r10	T	A	P	1
r10	T	geneR	P	1
r11	T	mrnaR	P	1
r12	T	mrnaR	P	1
r13	T	R	P	1
r16	T	R	P	1
r2	T	A	P	1
r2	T	geneA	P	1
r3	T	mrnaA	P	1
r4	T	mrnaA	P	1
r5	T	A	P	1
r8	T	A_R	P	1
r9	T	geneR_A	P	1

3.6 Read Arcs

Table 4: Read Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
geneA	P	r3	T	1
geneA_A	P	r4	T	1
geneR	P	r11	T	1
geneR_A	P	r12	T	1
mrnaA	P	r5	T	1
mrnaR	P	r13	T	1

3.7 Inhibitor Arcs

No elements available for description

3.8 Comments

Table 5: Comments Table

NET REFERENCE	COMMENT
TopLevel	PUR ORD HOM NBM CSV SCF FT0 TF0 FP0 PF0 CON SC NC N Y Y Y N N Y N Y Y Y N nES RKTH STP CPI CTI SCTI SB k-B 1-B DCF DSt DTr LIV REV N Y N Y N N - - - 0 - - - SSI RNK SCCS SECS Y 7 12 16

3.9 Images

No elements available for description

4 Declarations

This section contains information related to declarations specific to the net.

4.1 Constants

No elements available for description

4.2 Functions

No elements available for description

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.1.3 Erk_rg.sprgraph (Reachability Graph)

This is a single level net. The top level for this file is shown in Figure A.41. Customizations done in Snoopy2LaTeX dialog:

- **Declarations** and **Hierarchy** deselected for export (Figure A.42)
- For **Edges**, 'Transition' attribute deselected for export

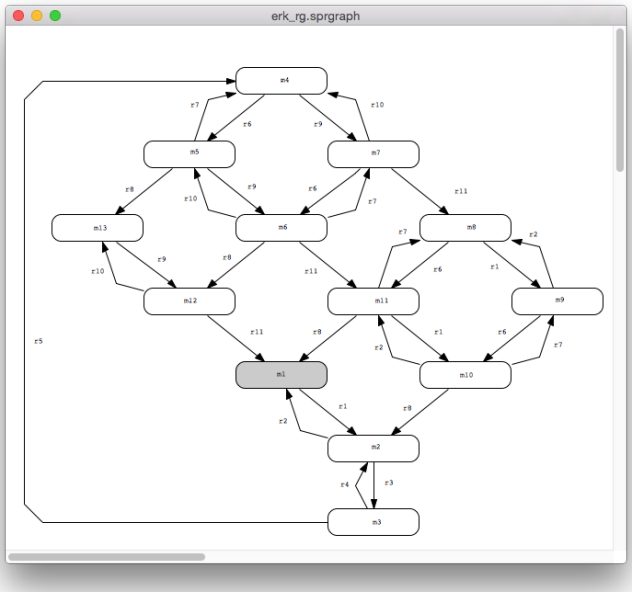


Figure A.41: Erk_rg - Snoopy file

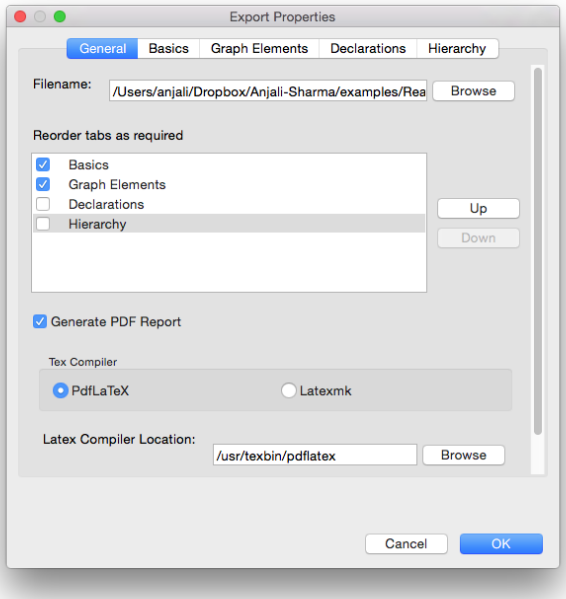
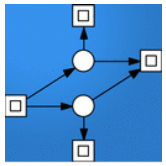


Figure A.42: Custom ordering in Snoopy2LaTeX

The following pages show the generated PDF report.



erk_rg.sprgraph

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

erk_rg.sprgraph Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 15:30

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: erk_rg.sprgraph
Creation Timestamp: 2015-06-17 15:28:37

1.2 Net Informations

Net Class: Reachability Graph

Element	Count
Node	13
Edge	35

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Nodes

Table 1: Nodes Table

NAME	ID	PLACES	CROSS REFS.
_N_0_	0		TopLevel
_N_1_	1		TopLevel
_N_2_	2		TopLevel
_N_3_	3		TopLevel
_N_4_	4		TopLevel
_N_5_	5		TopLevel
_N_6_	6		TopLevel
_N_7_	7		TopLevel
_N_8_	8		TopLevel
_N_9_	9		TopLevel
_N_10_	10		TopLevel
_N_11_	11		TopLevel
_N_12_	12		TopLevel

2.2 Edges

Table 2: Edges Table

SOURCE		TARGET	
NAME	TYPE	NAME	TYPE
_N_0_	N	_N_1_	N
_N_10_	N	_N_0_	N
_N_10_	N	_N_10_	N
_N_10_	N	_N_10_	N
_N_10_	N	_N_10_	N
_N_10_	N	_N_11_	N
_N_11_	N	_N_10_	N
_N_12_	N	_N_7_	N
_N_12_	N	_N_9_	N
_N_1_	N	_N_0_	N
_N_1_	N	_N_2_	N
_N_2_	N	_N_1_	N

Continued on next page

Table 2: Edges Table

SOURCE		TARGET	
NAME	TYPE	NAME	TYPE
_N_2_	N	_N_3_	N
_N_3_	N	_N_3_	N
_N_3_	N	_N_3_	N
_N_3_	N	_N_4_	N
_N_3_	N	_N_6_	N
_N_4_	N	_N_11_	N
_N_4_	N	_N_3_	N
_N_4_	N	_N_5_	N
_N_5_	N	_N_10_	N
_N_5_	N	_N_4_	N
_N_5_	N	_N_6_	N
_N_5_	N	_N_9_	N
_N_6_	N	_N_12_	N
_N_6_	N	_N_3_	N
_N_6_	N	_N_5_	N
_N_7_	N	_N_12_	N
_N_7_	N	_N_8_	N
_N_8_	N	_N_1_	N
_N_8_	N	_N_7_	N
_N_8_	N	_N_9_	N
_N_9_	N	_N_0_	N
_N_9_	N	_N_12_	N
_N_9_	N	_N_8_	N

2.3 Comments

No elements available for description

2.4 Images

No elements available for description

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

N Node

A.7.1.4 (Music Petri Net)

A.7.2 Quantitative Nets

A.7.2.1 Celegans.spstochpn (Stochastic Petri Net)

This is a single-level but large net. A partial view for this net file is shown in Figure A.43. Customizations done in Snoopy2LaTeX dialog:

- **Places** selected against regex - "active|AC" (Figure A.44)
- **Transition** selected against regex - "_7"
- All **Arcs**, **Lookup Table**, **Comments** and **Images** deselected for export (Figure A.44)
- **Generate Hierarchy Tree** deselected for export

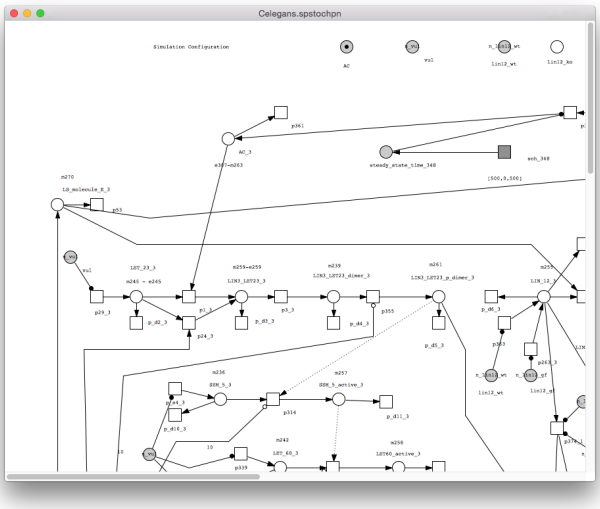


Figure A.43: Celegans - Snoopy file (Partial view due to large net structure)

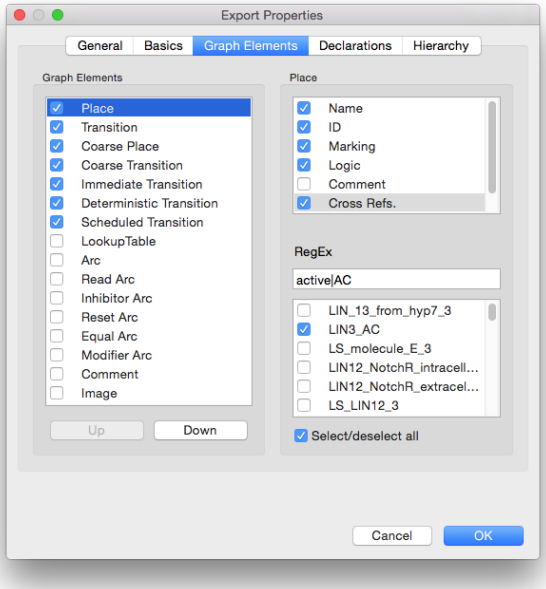
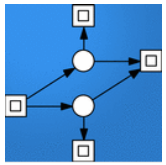


Figure A.44: Custom Graph Elements in Snoopy2LaTeX

The following pages show the generated PDF report.



Celegans.spstochpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

Celegans.spstochpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 16:22

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: Celegans.spstochpn
Creation Timestamp: 2010-11-03 10:07:50

1.2 Net Informations

Net Class: Stochastic Petri Net

Element	Count
Place	191
Transition	370
Immediate Transition	0
Deterministic Transition	0
Scheduled Transition	4
LookupTable	0
Arc	539
Read Arc	73
Inhibitor Arc	54
Reset Arc	0
Equal Arc	0
Modifier Arc	54

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
LIN3_AC	1	0	✗	TopLevel
SEM_5_active_3	12	0	✗	TopLevel
LET60_active_3	13	0	✗	TopLevel
MPK_1_active_C_3	17	0	✗	TopLevel
MPK_1_active_N_3	29	0	✗	TopLevel
MPK_1_active_N_4	30	0	✗	TopLevel
MPK_1_active_C_4	42	0	✗	TopLevel
LET60_active_4	46	0	✗	TopLevel
SEM_5_active_4	47	0	✗	TopLevel
SEM_5_active_5	70	0	✗	TopLevel
LET60_active_5	71	0	✗	TopLevel
MPK_1_active_C_5	75	0	✗	TopLevel
MPK_1_active_N_5	87	0	✗	TopLevel
MPK_1_active_N_8	88	0	✗	TopLevel
MPK_1_active_C_8	100	0	✗	TopLevel
LET60_active_8	104	0	✗	TopLevel
SEM_5_active_8	105	0	✗	TopLevel
SEM_5_active_7	128	0	✗	TopLevel
LET60_active_7	129	0	✗	TopLevel
MPK_1_active_C_7	133	0	✗	TopLevel
MPK_1_active_N_7	145	0	✗	TopLevel
MPK_1_active_N_6	146	0	✗	TopLevel
MPK_1_active_C_6	158	0	✗	TopLevel
LET60_active_6	162	0	✗	TopLevel
SEM_5_active_6	163	0	✗	TopLevel
AC_3	175	0	✗	TopLevel
AC_4	176	0	✗	TopLevel
AC_5	177	0	✗	TopLevel
AC_7	178	0	✗	TopLevel
AC_8	179	0	✗	TopLevel
AC	180	1	✓	TopLevel

2.2 Transitions

Table 2: Transitions Table

NAME	ID	LOG.			CROSS REFS.
p_d26_7	226	✗	*Main*	$MassAction(0.1)$	TopLevel
p_s9_7	227	✗	*Main*	0.05	TopLevel
p53_7	228	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d25_7	229	✗	*Main*	$MassAction(k2)$	TopLevel
p_d5_7	230	✗	*Main*	$MassAction(k5)$	TopLevel
p_d9_7	231	✗	*Main*	$MassAction(k12)$	TopLevel
p_d8_7	232	✗	*Main*	$MassAction(k13)$	TopLevel
p_d6_7	233	✗	*Main*	$MassAction(0.1)$	TopLevel
p19_7	234	✗	*Main*	$MassAction(k16)$	TopLevel
p_d4_7	235	✗	*Main*	$MassAction(k17)$	TopLevel
p_d3_7	236	✗	*Main*	$MassAction(0.1)$	TopLevel
p24_7	237	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d2_7	238	✗	*Main*	$MassAction(0.1)$	TopLevel
p_s2_7	239	✗	*Main*	1.0	TopLevel
p5_7	240	✗	*Main*	$MassAction(0.1)$	TopLevel
p4_7	241	✗	*Main*	$MassAction(0.1)$	TopLevel
p3_7	242	✗	*Main*	$MassAction(0.1 * medium)$	TopLevel
p6_7	243	✗	*Main*	$MassAction(0.1 * LIN3\_LET23\_p\_dimer\_7)$	TopLevel
p_s4_7	244	✗	*Main*	1	TopLevel
p_d10_7	245	✗	*Main*	$MassAction(k27)$	TopLevel
p_d11_7	246	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d13_7	247	✗	*Main*	$MassAction(k29)$	TopLevel
p_d12_7	248	✗	*Main*	$MassAction(k30)$	TopLevel
p_s5_7	249	✗	*Main*	$k31$	TopLevel
p7_7	250	✗	*Main*	$MassAction(0.1 * SEM\_5\_active\_7)$	TopLevel
pd14_7	251	✗	*Main*	$MassAction(k33)$	TopLevel
_T_252_p_s6_7	252	✗	*Main*	$k34$	TopLevel
p8_7	253	✗	*Main*	$MassAction(0.1 * LET60\_active\_7)$	TopLevel
p_d15_7	254	✗	*Main*	$MassAction(k37)$	TopLevel
p12_7	255	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d28_7	256	✗	*Main*	$MassAction(0.1)$	TopLevel
p374_7_1	257	✗	*Main*	$MassAction(1)$	TopLevel
p9_7	258	✗	*Main*	$MassAction(0.1)$	TopLevel
p22_7	259	✗	*Main*	$0.1 * mRNA\_of\_lst\_genes\_7$	TopLevel

Continued on next page

Table 2: Transitions Table

NAME	ID	LOG.			CROSS REFS.
p_d24_7	260	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d22_7	261	✗	*Main*	$MassAction(0.1)$	TopLevel
p_s8_7	262	✗	*Main*	0.1	TopLevel
p20_7	263	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d23_7	264	✗	*Main*	$MassAction(0.1)$	TopLevel
p21_7	265	✗	*Main*	$0.1 * LIN12\_NotchR\_intracellular\_N\_7$	TopLevel
p_d19_7	266	✗	*Main*	$MassAction(0.1)$	TopLevel
p_d16_7	267	✗	*Main*	$MassAction(0.1)$	TopLevel
p_u1_7	268	✗	*Main*	$MassAction(0.1)$	TopLevel
p11_7	269	✗	*Main*	$MassAction(0.1)$	TopLevel
p10_7	270	✗	*Main*	$0.1 * MPK\_1\_active\_N\_7$	TopLevel
pd21_7	271	✗	*Main*	$MassAction(k53)$	TopLevel
p15_7	272	✗	*Main*	$0.1 * Vulval\_gene\_7$	TopLevel
pd20_7	273	✗	*Main*	$MassAction(k55)$	TopLevel
ps7_7	274	✗	*Main*	0.1	TopLevel
pd18_7	275	✗	*Main*	$MassAction(0.1)$	TopLevel
p14_7	276	✗	*Main*	$0.1 * LIN\_31\_a\_7$	TopLevel
pd17_7	277	✗	*Main*	$MassAction(k59)$	TopLevel
p13_7	278	✗	*Main*	$MassAction(k60 * MPK\_1\_active\_N\_7)$	TopLevel
p3_u2_7	279	✗	*Main*	$MassAction(k62)$	TopLevel
p_s6_7	280	✗	*Main*	$MassAction(1)$	TopLevel
P57_7	281	✗	*Main*	$MassAction(LS\_Pink)$	TopLevel
p58_7	350	✗	*Main*	$MassAction(1)$	TopLevel
p59_7	351	✗	*Main*	$MassAction(1)$	TopLevel
p263_7	362	✗	*Main*	$MassAction(2)$	TopLevel
p374_7_2	368	✗	*Main*	$MassAction(1)$	TopLevel

2.3 Coarse Places

No elements available for description

2.4 Coarse Transitions

No elements available for description

2.5 Immediate Transitions

No elements available for description

2.6 Deterministic Transitions

No elements available for description

2.7 Scheduled Transitions

Table 3: Scheduled Transitions Table

NAME	ID	LOG.	PERIODIC LIST	BEGIN ; REPETITION ; END	CROSS REFS.
schedule_AC	370	✗	*Main*	400; 0; 400	TopLevel
sch_348	371	✗	*Main*	500; 0; 500	TopLevel
sch_57	372	✗	*Main*	450; 0; 450	TopLevel
sch_hyp7	373	✗	*Main*	200; 0; 200	TopLevel

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

Table 4: Constants Table

ID	GROUP	TYPE	NAME	MAIN				
0	marking	int	n_vul	1		0		
1	marking	int	n_lin12_wt	1				0
2	marking	int	n_lin12_gf	0				1
3	marking	int	n_lst	1	0	0	0	
4	marking	int	n_lin15	1			0	
5	parameter	double	k1	0.1				
6	parameter	double	k2	0.1				
7	parameter	double	k3	0.1				
8	parameter	double	k4	0.1				
9	parameter	double	k5	0.1				
10	parameter	double	k6	0.1				
11	parameter	double	k7	1				
12	parameter	double	k8	1				
13	parameter	double	k9	0.1				
14	parameter	double	k10	1				
15	parameter	double	k20	0.1				
16	parameter	double	k19	0.1				
17	parameter	double	k18	0.1				
18	parameter	double	k17	0.1				
19	parameter	double	k16	0.1				
20	parameter	double	k15	1				
21	parameter	double	k14	0.1				
22	parameter	double	k13	0.1				
23	parameter	double	k12	0.1				
24	parameter	double	k11	1				
25	parameter	double	k30	0.1				
26	parameter	double	k29	0.1				
27	parameter	double	k28	0.1				
28	parameter	double	k27	0.1				
29	parameter	double	k26	1				
30	parameter	double	k25	0.1				
31	parameter	double	k24	0.1				
32	parameter	double	k23	0.1				
33	parameter	double	k22	0.1				

Continued on next page

Table 4: Constants Table

ID	GROUP	TYPE	NAME	MAIN				
34	parameter	double	k21	1				
35	parameter	double	k40	0.1				
36	parameter	double	k39	0.1				
37	parameter	double	k38	0.1				
38	parameter	double	k37	0.1				
39	parameter	double	k36	0.1				
40	parameter	double	k35	0.1				
41	parameter	double	k34	1				
42	parameter	double	k33	0.1				
43	parameter	double	k32	0.1				
44	parameter	double	k31	1				
45	parameter	double	k50	0.1				
46	parameter	double	k49	0.1				
47	parameter	double	k48	0.1				
48	parameter	double	k47	0.1				
49	parameter	double	k46	0.1				
50	parameter	double	k45	0.1				
51	parameter	double	k44	1				
52	parameter	double	k43	0.1				
53	parameter	double	k42	0.1				
54	parameter	double	k41	0.1				
55	parameter	double	k60	0.1				
56	parameter	double	k59	0.1				
57	parameter	double	k58	0.1				
58	parameter	double	k57	0.1				
59	parameter	double	k56	1				
60	parameter	double	k55	0.1				
61	parameter	double	k54	0.1				
62	parameter	double	k53	0.1				
63	parameter	double	k52	0.1				
64	parameter	double	k51	0.1				
65	parameter	double	low	0.01				
66	parameter	double	medium	1				
67	parameter	double	high	100				
68	parameter	double	k61	2				
69	parameter	double	k62	0.1				
70	parameter	double	k63	0				
71	parameter	double	LS_Pink	0.1				
72	parameter	double	LS_Blue	1				
73	parameter	double	LS_Green	1				
74	parameter	double	_C_74_	1				

3.2 Functions

No elements available for description

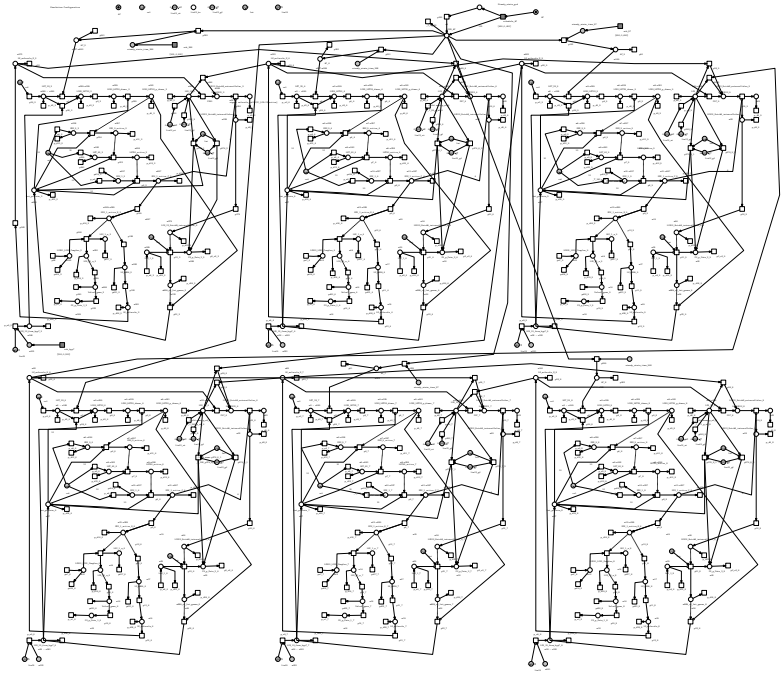
4 Hierarchy

This section contains information about the net hierarchy.

4.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

A.7.2.2 Circadian_rhythm.spcontped (Continuous Petri Net)

This is a single level net. The top level for this file is shown in Figure A.45. Customizations done in Snoopy2LaTeX dialog:

- **Hierarchy** deselected for export
- For **Coarse transition**, 'ID' attribute deselected for export

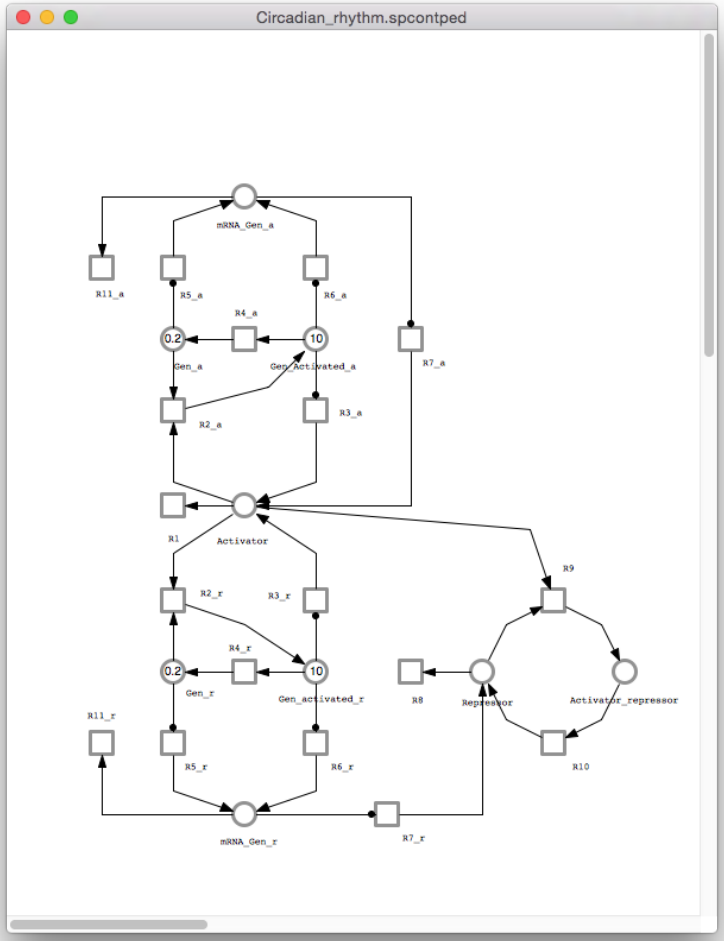
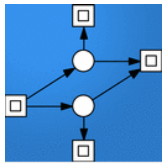


Figure A.45: Circadian_rhythm - Snoopy file

The following pages show the generated PDF report.



Circadian_rhythm.spcontped

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

Circadian_rhythm.spcontped Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 17:10

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: Circadian_rhythm.spcontped
Creation Timestamp: 2015-06-17 17:08:53

1.2 Net Informations

Net Class: Continuous Petri Net

Element	Count
Continuous Place	9
Continuous Transition	18
Arc	27
Inhibitor Arc	0
Test Arc	8
Modifier Arc	0

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Continuous Places

Table 1: Continuous Places Table

NAME	ID	MAR.	FIXED	LOG.	CROSS REFS.
Gen_a	0	0.2	0	✗	TopLevel
Gen_activated_r	1	10	0	✗	TopLevel
Gen_Activated_a	2	10	0	✗	TopLevel
Activator	3	0	0	✗	TopLevel
Gen_r	4	0.2	0	✗	TopLevel
mRNA_Gen_a	5	0	0	✗	TopLevel
Activator_repressor	6	0	0	✗	TopLevel
Repressor	7	0	0	✗	TopLevel
mRNA_Gen_r	8	0	0	✗	TopLevel

2.2 Continuous Transitions

Table 2: Continuous Transitions Table

NAME	FUNCTION SET	FUNCTION	LOG.	RE-VERSIBLE	CROSS REFS.
R4_a	*Main*	$MassAction(c1)$	✗	0	TopLevel
R6_a	*Main*	$GenA\_Activated * c5$	✗	0	TopLevel
R2_a	*Main*	$MassAction(c2)$	✗	0	TopLevel
R4_r	*Main*	$MassAction(c3)$	✗	0	TopLevel
R2_r	*Main*	$MassAction(c4)$	✗	0	TopLevel
R5_a	*Main*	$GenA * c6$	✗	0	TopLevel
R11_a	*Main*	$MassAction(10)$	✗	0	TopLevel
R3_a	*Main*	$GenA\_Activated * c1$	✗	0	TopLevel

Continued on next page

Table 2: Continuous Transitions Table

NAME	FUNCTION SET	FUNCTION	LOG.	RE-VERSIBLE	CROSS REFS.
R3_r	*Main*	$GenB\_activated * c3$	✗	0	TopLevel
R7_a	*Main*	$mRNA\_GenA * c8$	✗	0	TopLevel
R1	*Main*	$MassAction(c11)$	✗	0	TopLevel
R9	*Main*	$MassAction(1)$	✗	0	TopLevel
R5_r	*Main*	$GenB * c14$	✗	0	TopLevel
R6_r	*Main*	$GenB\_activated * c13$	✗	0	TopLevel
R11_r	*Main*	$MassAction(c15)$	✗	0	TopLevel
R7_r	*Main*	$mRNA\_GenB * c16$	✗	0	TopLevel
R8	*Main*	$MassAction(c17)$	✗	0	TopLevel
R10	*Main*	$MassAction(c2)$	✗	0	TopLevel

2.3 Coarse Places

No elements available for description

2.4 Coarse Transitions

No elements available for description

Continued on next page

Table 3: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.

2.5 Arcs

Table 3: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
Activator	PC	R1	TC	1
Activator	PC	R2_a	TC	1
Activator	PC	R2_r	TC	1
Activator	PC	R9	TC	1
Activator_repressor	PC	R10	TC	1
Gen_Activated_a	PC	R4_a	TC	1
Gen_a	PC	R2_a	TC	1
Gen_activated_r	PC	R4_r	TC	1
Gen_r	PC	R2_r	TC	1
Repressor	PC	R8	TC	1
Repressor	PC	R9	TC	1
mRNA_Gen_a	PC	R11_a	TC	1
mRNA_Gen_r	PC	R11_r	TC	1
R10	TC	Repressor	PC	1
R2_a	TC	Gen_Activated_a	PC	1
R2_r	TC	Gen_activated_r	PC	1
R3_a	TC	Activator	PC	1
R3_r	TC	Activator	PC	1
R4_a	TC	Gen_a	PC	1
R4_r	TC	Gen_r	PC	1
R5_a	TC	mRNA_Gen_a	PC	1
R5_r	TC	mRNA_Gen_r	PC	1
R6_a	TC	mRNA_Gen_a	PC	1
R6_r	TC	mRNA_Gen_r	PC	1
R7_a	TC	Activator	PC	1
R7_r	TC	Repressor	PC	1
R9	TC	Activator_repressor	PC	1

2.6 Inhibitor Arcs

Table 4: Inhibitor Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.

2.7 Test Arcs

Table 5: Test Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
Gen_Activated_a	PC	R3_a	TC	1
Gen_Activated_a	PC	R6_a	TC	1
Gen_a	PC	R5_a	TC	1
Gen_activated_r	PC	R3_r	TC	1
Gen_activated_r	PC	R6_r	TC	1
Gen_r	PC	R5_r	TC	1
mRNA_Gen_a	PC	R7_a	TC	1
mRNA_Gen_r	PC	R7_r	TC	1

2.8 Modifier Arcs

No elements available for description

2.9 Comments

No elements available for description

2.10 Images

No elements available for description

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

Table 6: Constants Table

NAME	ID	GROUP	TYPE	MAIN
c1	0	parameter	double	50
c2	1	parameter	double	1
c3	2	parameter	double	100
c4	3	parameter	double	1
c5	4	parameter	double	50
c6	5	parameter	double	0.01
c7	6	parameter	double	10
c8	7	parameter	double	50
c11	8	parameter	double	1
c12	9	parameter	double	2
c13	10	parameter	double	50
c14	11	parameter	double	0.01
c15	12	parameter	double	0.5
c16	13	parameter	double	5
c17	14	parameter	double	0.2
SegmaR	15	parameter	double	0.2

3.2 Functions

No elements available for description

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

PC Place, Continuous

TC Transition, Continuous

A.7.2.3 EucaryoticCell.sphybrid (Hybrid Petri Net)

This is a single level net. The top level for this file is shown in Figure A.46. Customizations done in Snoopy2LaTeX dialog:

- All arc types except **Arc**, **Comment** and **Images** deselected for export
- **Constants** selected against regex - "Kcm|K*r"

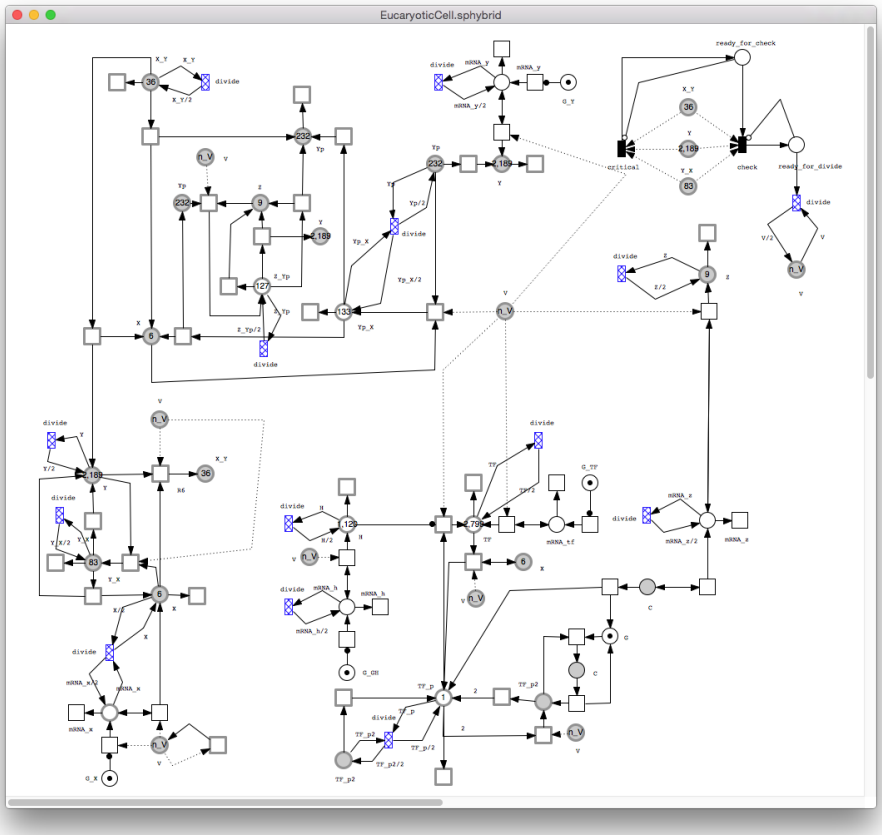
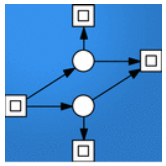


Figure A.46: EucaryoticCell - Snoopy file

The following pages show the generated PDF report.



EucaryoticCell.sphybrid

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

EucaryoticCell.sphybrid Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 17:33

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: EucaryoticCell.sphybrid
Creation Timestamp: 2012-03-29 14:40:53

1.2 Net Informations

Net Class: Hybrid Petri Net

Element	Count
Discrete Place	12
Continuous Place	14
Stochastic Transition	18
Continuous Transition	30
Immediate Transition	3
Deterministic Transition	0
Scheduled Transition	0
Arc	135
Read Arc	5
Inhibitor Arc	2
Reset Arc	0
Equal Arc	0
Modifier Edge	20

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Discrete Places

Table 1: Discrete Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
mRNA_z	0	0	✗	TopLevel
mRNA_y	1	0	✗	TopLevel
C	2	0	✓	TopLevel
mRNA_h	3	0	✗	TopLevel
mRNA_tf	4	0	✗	TopLevel
G_X	5	1	✗	TopLevel
G_Y	6	1	✗	TopLevel
G_TF	7	1	✗	TopLevel
G_GH	8	1	✗	TopLevel
ready_for_divide	9	0	✗	TopLevel
G	10	1	✗	TopLevel
ready_for_check	11	0	✗	TopLevel

2.2 Continuous Places

Table 2: Continuous Places Table

FIXED	NAME	ID	MAR.	LOG.	CROSS REFS.
0	V	12	n_V	✓	TopLevel
0	Y	13	2189	✓	TopLevel
0	Yp	14	232	✓	TopLevel
0	Z	15	9	✓	TopLevel
0	Yp_X	16	133	✗	TopLevel
0	X_Y	17	36	✓	TopLevel
0	X	18	6	✓	TopLevel
0	H	19	1120	✗	TopLevel
0	TF	20	2799	✗	TopLevel
0	Y_X	21	83	✓	TopLevel
0	Z_Yp	22	127	✗	TopLevel
0	TF_p2	23	0	✓	TopLevel
0	TF_p	24	1	✗	TopLevel
0	mRNA_x	25	0	✗	TopLevel

2.3 Stochastic Transitions

Table 3: Stochastic Transitions Table

NAME	ID	LOG.	FUNCTION SET	FUNCTION	CROSS REFS.
R15	0	✗	$Main$	$MassAction(Ksz*V)$	TopLevel
_T_1_	1	✗	$Main$	$MassAction(Ksh*V)$	TopLevel
_T_2_	2	✗	$Main$	$MassAction(Kdmx)$	TopLevel
_T_3_	3	✗	$Main$	$MassAction(Ksmz)$	TopLevel
R25	4	✗	$Main$	$MassAction(Kdmz)$	TopLevel
R26	5	✗	$Main$	$MassAction(Ksf*V)$	TopLevel
R39	6	✗	$Main$	$MassAction(Ksmx*V)$	TopLevel
_T_7_	7	✗	$Main$	$MassAction(Ksmy)$	TopLevel
R42	8	✗	$Main$	$MassAction(Kdmy)$	TopLevel
R43	9	✗	$Main$	$MassAction(Ksmf)$	TopLevel
R44	10	✗	$Main$	$MassAction(Kdmf)$	TopLevel
R45	11	✗	$Main$	$MassAction(Ksmh)$	TopLevel
R46	12	✗	$Main$	$MassAction(Kdmh)$	TopLevel
R47	13	✗	$Main$	$MassAction(2*Kdf)$	TopLevel
R32	14	✗	$Main$	$MassAction(Ksy*V)$	TopLevel
_T_15_	15	✗	$Main$	$MassAction(Kcf)$	TopLevel
R22	16	✗	$Main$	$MassAction(Kcr)$	TopLevel

Continued on next page

Table 3: Stochastic Transitions Table

NAME	ID	LOG.	FUNCTION SET	FUNCTION	CROSS REFS.
R1	17	✗	*Main*	$MassAction(Ksx*V)$	TopLevel

2.4 Continuous Transitions

Table 4: Continuous Transitions Table

NAME	ID	LOG.	FUNCTION SET	FUNCTION	CROSS REFS.
_T_18_	18	✗	*Main*	$mu*V$	TopLevel
K33	19	✗	*Main*	$MassAction(Kdy)$	TopLevel
R23	20	✗	*Main*	$MassAction(Kby)$	TopLevel
R9	21	✗	*Main*	$MassAction(Kzypf/V)$	TopLevel
R34	22	✗	*Main*	$MassAction(Kdy)$	TopLevel
R14	23	✗	*Main*	$MassAction(Kypx\_cat)$	TopLevel
R37	24	✗	*Main*	$MassAction(Kdy)$	TopLevel
R12	25	✗	*Main*	$MassAction(Kypxf/V)$	TopLevel
_T_26_	26	✗	*Main*	$MassAction(Kdz)$	TopLevel
R13	27	✗	*Main*	$MassAction(Kypxr)$	TopLevel
KR7	28	✗	*Main*	$MassAction(Kxyr)$	TopLevel
R31	29	✗	*Main*	$MassAction(Kdh)$	TopLevel
R27	30	✗	*Main*	$MassAction(Kdf)$	TopLevel
R17	31	✗	*Main*	$MassAction(Kpf/V)$	TopLevel
R6	32	✗	*Main*	$MassAction(Kxyf/V)$	TopLevel
R18	33	✗	*Main*	$MassAction(Kpr/V)$	TopLevel
R29	34	✗	*Main*	$MassAction(2*Kdf)$	TopLevel

Continued on next page

Table 4: Continuous Transitions Table

NAME	ID	LOG.	FUNCTION SET	FUNCTION	CROSS REFS.
R5	35	✗	*Main*	$MassAction(Kyx\_cat)$	TopLevel
R35	36	✗	*Main*	$MassAction(Kdy)$	TopLevel
R8	37	✗	*Main*	$MassAction(Kxy\_cat)$	TopLevel
R36	38	✗	*Main*	$MassAction(Kdy)$	TopLevel
R2	39	✗	*Main*	$MassAction(Kdx)$	TopLevel
R4	40	✗	*Main*	$MassAction(Kyxr)$	TopLevel
R3	41	✗	*Main*	$MassAction(Kyxf/V)$	TopLevel
R38	42	✗	*Main*	$MassAction(Kdy)$	TopLevel
R11	43	✗	*Main*	$MassAction(Kzyp\_cat)$	TopLevel
R10	44	✗	*Main*	$MassAction(Kzypr)$	TopLevel
R19	45	✗	*Main*	$MassAction(Kdim/V)$	TopLevel
R20	46	✗	*Main*	$MassAction(Kdiss)$	TopLevel
R28	47	✗	*Main*	$MassAction(Kdf)$	TopLevel

2.5 Immediate Transitions

Table 5: Immediate Transitions Table

NAME	ID	LOG.	WEIGHT SET	WEIGHT	CROSS REFS.
check	48	✗	*Main*	$gt(Y+Y\_X+X\_Y, Threshold)$	TopLevel
divide	49	✓	*Main*	10	TopLevel
critical	50	✗	*Main*	$lt(Y+Y\_X+X\_Y, Critical)$	TopLevel

2.6 Deterministic Transitions

No elements available for description

2.7 Scheduled Transitions

No elements available for description

2.8 Coarse Places

No elements available for description

2.9 Coarse Transitions

No elements available for description

2.10 Arcs

Table 6: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
C	P	R22	T	1
C	P	R47	T	1
C	P	_T_3_	T	1
G	P	_T_15_	T	1
H	P	R31	T	1
H	P	divide	I	H
TF	P	R17	T	1
TF	P	R27	T	1
TF	P	divide	I	TF
TF_p	P	R18	T	1
TF_p	P	R19	T	2
TF_p	P	R28	T	1
TF_p	P	divide	I	TF_p
TF_p2	P	R20	T	1
TF_p2	P	R29	T	1
TF_p2	P	_T_15_	T	1
TF_p2	P	divide	I	TF_p2
V	P	divide	I	V

Continued on next page

Table 6: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
X	P	R12	T	1
X	P	R17	T	1
X	P	R2	T	1
X	P	R3	T	1
X	P	R6	T	1
X	P	divide	I	X/2
X_Y	P	KR7	T	1
X_Y	P	R35	T	1
X_Y	P	R8	T	1
X_Y	P	divide	I	X_Y
Y	P	K33	T	1
Y	P	R3	T	1
Y	P	R6	T	1
Y	P	divide	I	Y
Y_X	P	R38	T	1
Y_X	P	R4	T	1
Y_X	P	R5	T	1
Y_X	P	divide	I	Y_X
Yp	P	R12	T	1
Yp	P	R23	T	1
Yp	P	R34	T	1
Yp	P	R9	T	1
Yp	P	divide	I	Yp
Yp_X	P	R13	T	1
Yp_X	P	R14	T	1
Yp_X	P	R37	T	1
Yp_X	P	divide	I	Yp_X
Z	P	R9	T	1
Z	P	_T_26_	T	1
Z	P	divide	I	Z
Z_Yp	P	R10	T	1
Z_Yp	P	R11	T	1
Z_Yp	P	R36	T	1
Z_Yp	P	divide	I	Z_Yp
mRNA_h	P	R46	T	1
mRNA_h	P	_T_1_	T	1
mRNA_h	P	divide	I	mRNA_h
mRNA_tf	P	R26	T	1
mRNA_tf	P	R44	T	1
mRNA_x	P	R1	T	1

Continued on next page

Table 6: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
mRNA_x	P	_T_2_	T	1
mRNA_x	P	divide	I	mRNA_x
mRNA_y	P	R32	T	1
mRNA_y	P	R42	T	1
mRNA_y	P	divide	I	mRNA_y
mRNA_z	P	R15	T	1
mRNA_z	P	R25	T	1
mRNA_z	P	divide	I	mRNA_z
ready_for_check	P	check	I	1
ready_for_divide	P	divide	I	1
KR7	T	X	P	1
KR7	T	Y	P	1
R1	T	X	P	1
R1	T	mRNA_x	P	1
R10	T	Yp	P	1
R10	T	Z	P	1
R11	T	Y	P	1
R11	T	Z	P	1
R12	T	Yp_X	P	1
R13	T	X	P	1
R13	T	Yp	P	1
R14	T	Yp	P	1
R15	T	Z	P	1
R15	T	mRNA_z	P	1
R17	T	TF_p	P	1
R17	T	X	P	1
R18	T	TF	P	1
R19	T	TF_p2	P	1
R20	T	TF_p	P	2
R22	T	G	P	1
R22	T	TF_p2	P	1
R23	T	Y	P	1
R26	T	TF	P	1
R26	T	mRNA_tf	P	1
R29	T	TF_p	P	1
R3	T	Y_X	P	1
R32	T	Y	P	1
R32	T	mRNA_y	P	1
R36	T	Z	P	1
R39	T	mRNA_x	P	1

Continued on next page

Table 6: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
R4	T	X	P	1
R4	T	Y	P	1
R43	T	mRNA_tf	P	1
R45	T	mRNA_h	P	1
R47	T	G	P	1
R47	T	TF_p	P	1
R5	T	Y	P	1
R6	T	X_Y	P	1
R8	T	X	P	1
R8	T	Yp	P	1
R9	T	Z_Yp	P	1
_T_15_	T	C	P	1
_T_18_	T	V	P	1
_T_1_	T	H	P	1
_T_1_	T	mRNA_h	P	1
_T_3_	T	C	P	1
_T_3_	T	mRNA_z	P	1
_T_7_	T	mRNA_y	P	1
check	I	ready_for_divide	P	1
critical	I	ready_for_check	P	1
divide	I	H	P	H/2
divide	I	TF	P	TF/2
divide	I	TF_p	P	TF_p/2
divide	I	TF_p2	P	TF_p2/2
divide	I	V	P	V/2
divide	I	X	P	X
divide	I	X_Y	P	X_Y/2
divide	I	Y	P	Y/2
divide	I	Y_X	P	Y_X/2
divide	I	Yp	P	Yp/2
divide	I	Yp_X	P	Yp_X/2
divide	I	Z	P	Z/2
divide	I	Z_Yp	P	Z_Yp/2
divide	I	mRNA_h	P	mRNA_h/2
divide	I	mRNA_x	P	mRNA_x/2
divide	I	mRNA_y	P	mRNA_y/2
divide	I	mRNA_z	P	mRNA_z/2

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

Table 7: Constants Table

ID	GROUP	TYPE	NAME	MAIN	V-SET 1
2	parameter	double	Ksmx	1	
8	parameter	double	Ksmy	7	
15	parameter	double	Kyxr	12	
16	parameter	double	Kxyr	42	
19	parameter	double	Ksmz	1.35	
21	parameter	double	Ksmf	7	
23	parameter	double	Ksmh	7	
29	parameter	double	Kzypr	12	
30	parameter	double	Kypxr	12	
33	parameter	double	Kcr	2.7	
38	parameter	double	Kpr	0.03594	
39	parameter	double	Threshold	1200	
42	parameter	double	Critical	300	1

3.2 Functions

No elements available for description

4 Hierarchy

This section contains information about the net hierarchy.

4.1 Hierarchy Tree

This section contains the hierarchical tree structure for selected levels.

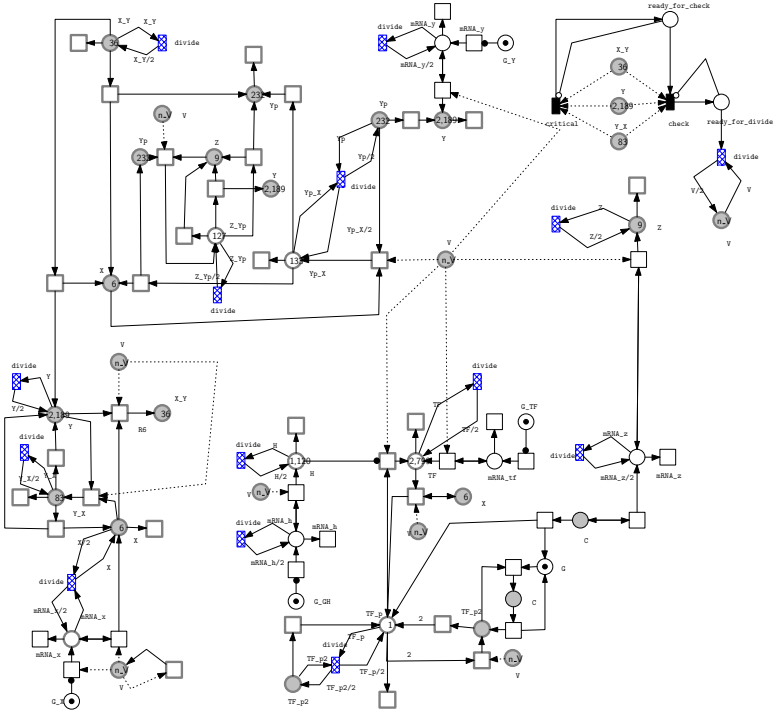
0. Top Level

0.TopLevel

4.2 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

I Immediate Transition

P Place

T Transition

A.7.2.4 Torch.sptpt (Time Petri Net)

This is a single level net. The top level for this file is shown in Figure A.47. Customizations done in Snoopy2LaTeX dialog:

- **Latexmk** selected for compilation
- **Landscape** orientation of generated report
- **Declarations** deselected for export
- **Coarse Place**, **Coarse Transition**, **Comment** and **Image** deselected for export (Figure A.48)
- **Arc** reordered to precede all other graph elements in report (Figure A.48)

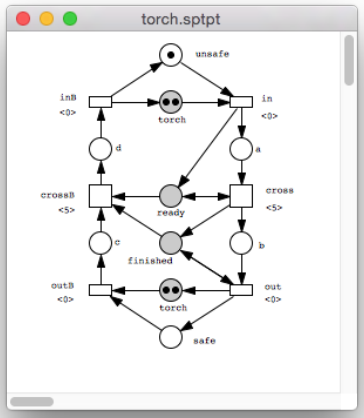


Figure A.47: Torch - Snoopy file

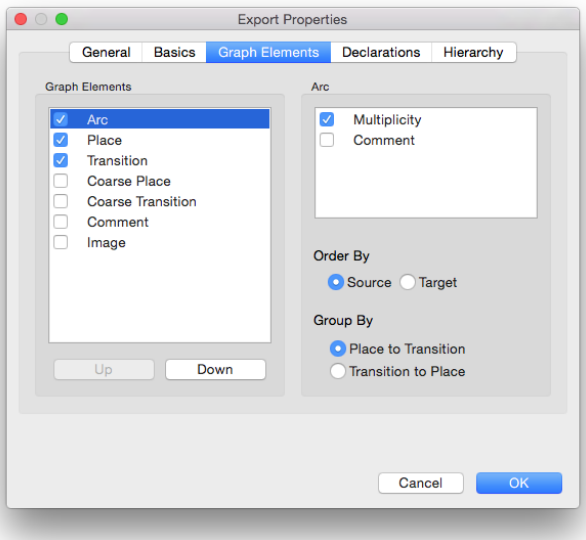
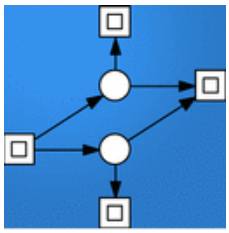


Figure A.48: Custom Graph Elements in Snoopy2LaTeX

The following pages show the generated PDF report.[1]

[1]Please note that this report is oriented to landscape mode.



torch.sptpt

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

torch.sptpt Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: torch.sptpt
Creation Timestamp: 2015-06-17 15:36:00

1.2 Net Informations

Net Class: Time Petri Net

Element	Count
Place	9
Transition	6
Arc	24

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Arcs

Table 1: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
a	P	cross	T	1
b	P	out	T	1
c	P	crossB	T	1
d	P	inB	T	1
finished	P	crossB	T	1
finished	P	out	T	1
ready	P	cross	T	1
ready	P	crossB	T	1
safe	P	outB	T	1
torch	P	in	T	1
torch	P	outB	T	1
unsafe	P	in	T	1
cross	T	b	P	1
cross	T	finished	P	1
cross	T	ready	P	1
crossB	T	d	P	1
in	T	a	P	1
in	T	ready	P	1

Continued on next page

Table 1: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
inB	T	torch	P	1
inB	T	unsafe	P	1
out	T	finished	P	1
out	T	safe	P	1
out	T	torch	P	1
outB	T	c	P	1

2.2 Places

Table 2: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
unsafe	0	1	✗	TopLevel
torch	1	2	✓	TopLevel
ready	2	0	✓	TopLevel
finished	3	0	✓	TopLevel
safe	4	0	✗	TopLevel
b	5	0	✗	TopLevel
c	6	0	✗	TopLevel
a	7	0	✗	TopLevel
d	8	0	✗	TopLevel

2.3 Transitions

Table 3: Transitions Table

NAME	ID	LOG.	DURATION LIST	DURATION	INTERVAL LIST	EFT ; LFT	CROSS REFS.
in	0	✗	*Main*	0	*Main*	?; ?	TopLevel
inB	1	✗	*Main*	0	*Main*	?; ?	TopLevel
cross	2	✗	*Main*	5	*Main*	?; ?	TopLevel
out	3	✗	*Main*	0	*Main*	?; ?	TopLevel
outB	4	✗	*Main*	0	*Main*	?; ?	TopLevel
crossB	5	✗	*Main*	5	*Main*	?; ?	TopLevel

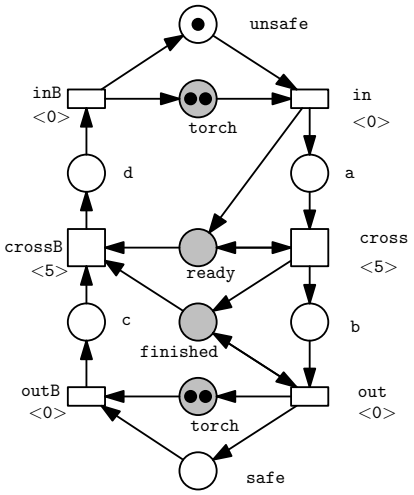
3 Hierarchy

This section contains information about the net hierarchy.

3.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.3 Qualitative Nets (Colored)

A.7.3.1 Philosopher.colpn (Colored Petri Net)

This is a single level net. The top level for this file is shown in Figure A.49. Customizations done in Snoopy2LaTeX dialog:

- **Basics** deselected for export
- **Declarations** reordered to precede 'Graph Elements' in report

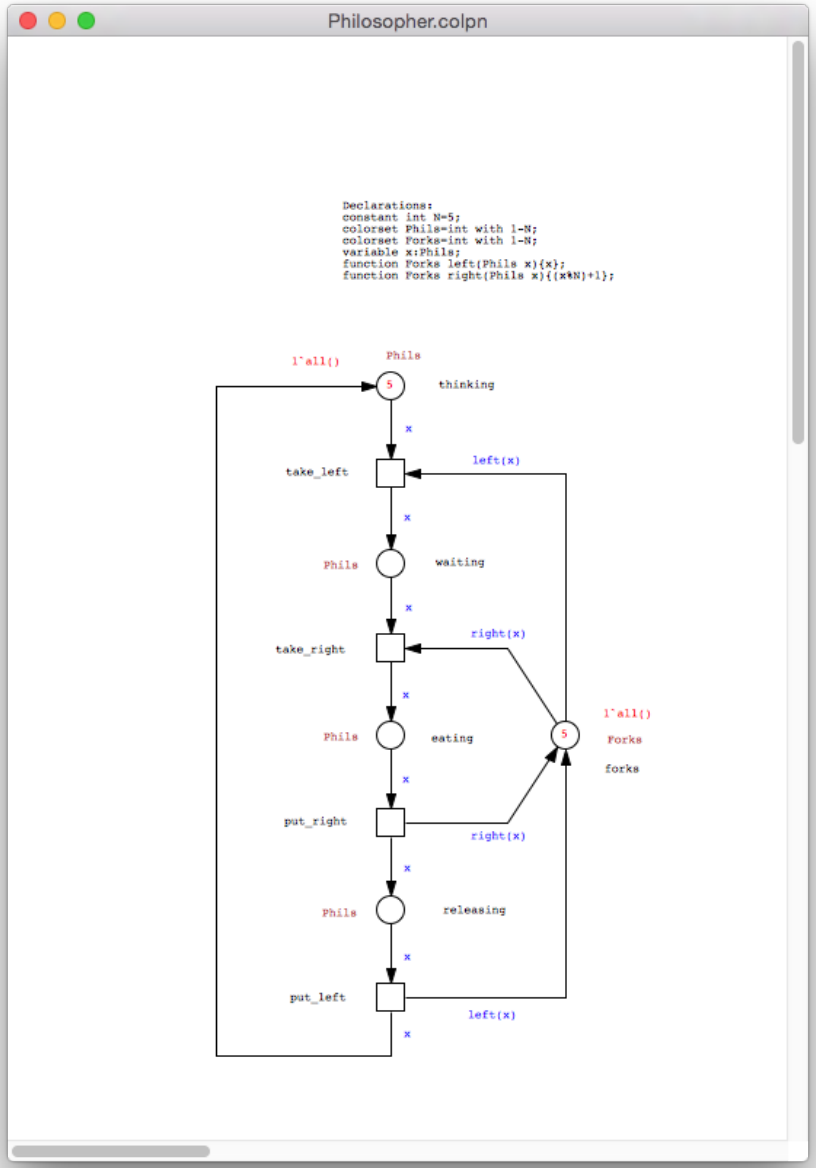
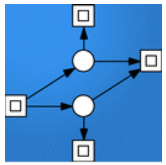


Figure A.49: Philosopher - Snoopy file

The following pages show the generated PDF report.



Philosopher.colpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

Philosopher.colpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 18:43

Contents

1 Declarations

This section contains information related to declarations specific to the net.

1.1 Constants

Table 1: Constants Table

NAME	TYPE	VALUE
N	*int*	5

1.2 Functions

Table 2: Functions Table

RETURN TYPE	FUNCTION NAME	PARAMETER LIST	FUNCTION BODY
Forks	*left*	*Philsx*	x
Forks	*right*	*Philsx*	$(x\%N)+1$

1.3 Simple Color Sets

Table 3: Simple Color Sets Table

NAME	TYPE	COLORS	PLACE COLOR
Dot	*dot*	*dot*	
Phils	*int*	$1-N$	
Forks	*int*	$1-N$	

1.4 Compound Color Sets

No elements available for description

1.5 Alias Color Sets

No elements available for description

1.6 Variables

Table 4: Variables Table

NAME	COLOR SET
x	*Phils*

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 5: Places Table

NAME	ID	MAR.	LOG.	COLOR/ /PREDICATE/ /FUNCTION	MARKING	COLORSET	CROSS REFS.
thinking	0	5	✗	*all*()	1	Phils	TopLevel
waiting	1	0	✗	*all*()	0	Phils	TopLevel
eating	2	0	✗	*all*()	0	Phils	TopLevel
releasing	3	0	✗	*all*()	0	Phils	TopLevel
forks	4	5	✗	*all*()	1	Forks	TopLevel

2.2 Transitions

Table 6: Transitions Table

NAME	ID	LOG.	GUARD SET	GUARD	CROSS REFS.
take_left	0	✗	*Main*		TopLevel
take_right	1	✗	*Main*		TopLevel
put_right	2	✗	*Main*		TopLevel
put_left	3	✗	*Main*		TopLevel

2.3 Coarse Places

No elements available for description

2.4 Coarse Transitions

No elements available for description

2.5 Arcs

Table 7: Arcs Table

SOURCE		TARGET		MUL.	EXPRES-SION SET	EXPRES-SION
NAME	TYPE	NAME	TYPE			
eating	P	put_right	T	1	*Main*	x
forks	P	take_left	T	1	*Main*	$left(x)$
forks	P	take_right	T	1	*Main*	$right(x)$
releasing	P	put_left	T	1	*Main*	x
thinking	P	take_left	T	1	*Main*	x
waiting	P	take_right	T	1	*Main*	x
put_left	T	forks	P	1	*Main*	$left(x)$
put_left	T	thinking	P	1	*Main*	x
put_right	T	forks	P	1	*Main*	$right(x)$
put_right	T	releasing	P	1	*Main*	x
take_left	T	waiting	P	1	*Main*	x
take_right	T	eating	P	1	*Main*	x

2.6 Comments

Table 8: Comments Table

NET REFERENCE	COMMENT
TopLevel	Declarations: constant int N=5; colorset Phils=int with 1-N; colorset Forks=int with 1-N; variable x:Phils; function Forks left(Phils x){x}; function Forks right(Phils x){(x%N)+1};

2.7 Images

No elements available for description

3 Hierarchy

This section contains information about the net hierarchy.

3.1 Hierarchy Tree

This section contains the hierarchical tree structure for selected levels.

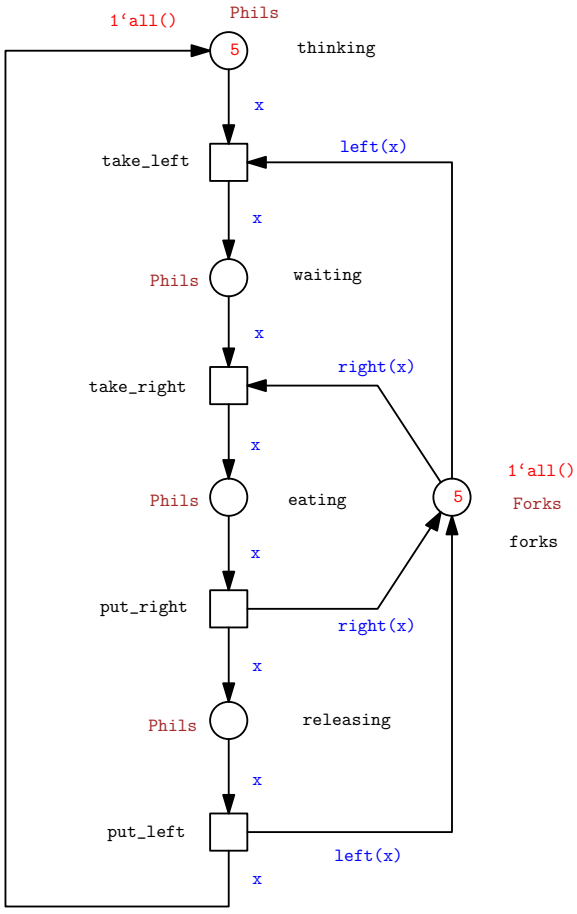
0. Top Level

3.2 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level

```
Declarations:
constant int N=5;
colorset Phils=int with 1-N;
colorset Forks=int with 1-N;
variable x:Phils;
function Forks left(Phils x){x};
function Forks right(Phils x){(x%N)+1};
```



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.3.2 Control_flow.colextpn (Colored Extended Petri Net)

This is a single level net. The top level for this file is shown in Figure A.50. No customizations are done in Snoopy2LaTeX dialog. Therefore, the report is generated based on the default settings in the dialog.

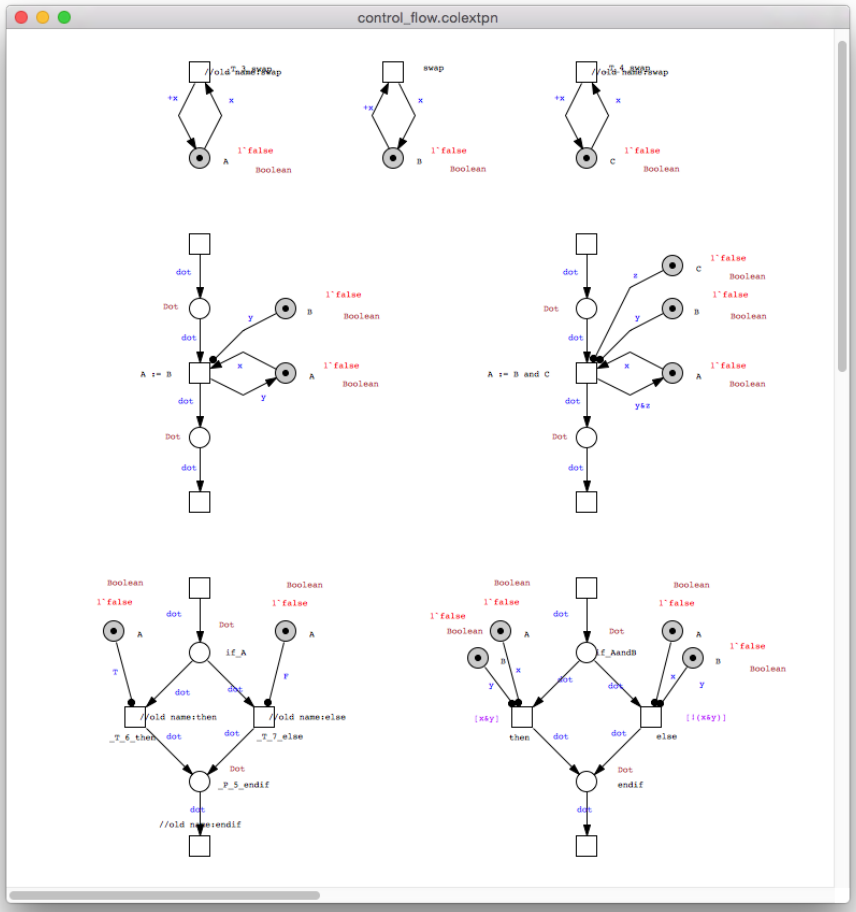
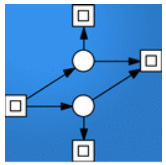


Figure A.50: Control Flow - Snoopy file

The following pages show the generated PDF report.



control_flow.colextpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

control_flow.colextpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 20:15

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: control_flow.colextpn
Creation Timestamp: 2011-02-03 00:12:06

1.2 Net Informations

Net Class: Colored Extended Petri Net

Element	Count
Place	11
Transition	17
Arc	30
Read Arc	9
Inhibitor Arc	0
Reset Arc	0
Equal Arc	0

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	COLOR/ /PREDICATE/ /FUNCTION	MARKING	COLORSET	CROSS REFS.
_P_0_	0	0	✗	*dot*	0	Dot	TopLevel
_P_1_	1	0	✗	*all*()	0	Dot	TopLevel
A	2	1	✓	*false*	1	Boolean	TopLevel
B	3	1	✓	*false*	1	Boolean	TopLevel
if_A	4	0	✗	*all*()	0	Dot	TopLevel
_P_5_endif	5	0	✗	*all*()	0	Dot	TopLevel
C	6	1	✓	*false*	1	Boolean	TopLevel
_P_7_	7	0	✗	*all*()	0	Dot	TopLevel
_P_8_	8	0	✗	*dot*	0	Dot	TopLevel
endif	9	0	✗	*all*()	0	Dot	TopLevel
if_AandB	10	0	✗	*all*()	0	Dot	TopLevel

2.2 Transitions

Table 2: Transitions Table

NAME	ID	LOG.	GUARD SET	GUARD	CROSS REFS.
_T_0_	0	✗	*Main*		TopLevel
_T_1_	1	✗	*Main*		TopLevel
_T_2_	2	✗	*Main*		TopLevel
_T_3_swap	3	✗	*Main*		TopLevel
_T_4_swap	4	✗	*Main*		TopLevel
_T_5_	5	✗	*Main*		TopLevel
_T_6_then	6	✗	*Main*		TopLevel
_T_7_else	7	✗	*Main*		TopLevel
_T_8_	8	✗	*Main*		TopLevel
swap	9	✗	*Main*		TopLevel

Continued on next page

Table 2: Transitions Table

NAME	ID	LOG.	GUARD SET	GUARD	CROSS REFS.
_T_10_	10	✗	*Main*		TopLevel
_T_11_	11	✗	*Main*		TopLevel
_T_12_	12	✗	*Main*		TopLevel
_T_13_	13	✗	*Main*		TopLevel
else	14	✗	*Main*	!(x&y)	TopLevel
then	15	✗	*Main*	x&y	TopLevel
_T_16_	16	✗	*Main*		TopLevel

2.3 Coarse Places

No elements available for description

2.4 Coarse Transitions

No elements available for description

2.5 Arcs

Table 3: Arcs Table

SOURCE		TARGET		MUL.	EXPRESSION SET	EXPRESSION
NAME	TYPE	NAME	TYPE			
A	P	_T_0_	T	1	*Main*	x
A	P	_T_12_	T	1	*Main*	x
A	P	_T_3_swap	T	1	*Main*	x
B	P	swap	T	1	*Main*	x
C	P	_T_4_swap	T	1	*Main* *false*	x 1
_P_0_	P	_T_0_	T	1	*Main*	*dot*
_P_1_	P	_T_2_	T	1	*Main*	*dot*
_P_5_endif	P	_T_8_	T	1	*Main* *dot*	*dot* 1
_P_7_	P	_T_10_	T	1	*Main*	*dot*
_P_8_	P	_T_12_	T	1	*Main*	*dot*
endif	P	_T_13_	T	1	*Main*	*dot*
if_A	P	_T_6_then	T	1	*Main*	*dot*

Continued on next page

Table 3: Arcs Table

SOURCE		TARGET		MUL.	EXPRESSION SET	EXPRESSION
NAME	TYPE	NAME	TYPE			
if_A	P	_T_7_else	T	1	*Main*	*dot*
if_AandB	P	else	T	1	*Main*	*dot*
if_AandB	P	then	T	1	*Main*	*dot*
_T_0_	T	A	P	1	*Main* *true*	*y* 1
_T_0_	T	_P_1_	P	1	*Main* *dot*	*dot* 1
_T_11_	T	_P_8_	P	1	*Main* *dot*	*dot* 1
_T_12_	T	A	P	1	*Main*	*y&z*
_T_12_	T	_P_7_	P	1	*Main*	*dot*
_T_16_	T	if_AandB	P	1	*Main* *dot*	*dot* 1
_T_1_	T	_P_0_	P	1	*Main* *dot*	*dot* 1
_T_3_swap	T	A	P	1	*Main* *true*	$+x$ 1
_T_4_swap	T	C	P	1	*Main* *true*	$+x$ 1
_T_5_	T	if_A	P	1	*Main* *dot*	*dot* 1
_T_6_then	T	_P_5_endif	P	1	*Main*	*dot*
_T_7_else	T	_P_5_endif	P	1	*Main* *dot*	*dot* 1
else	T	endif	P	1	*Main*	*dot*
swap	T	B	P	1	*Main* *false*	$+x$ 1
then	T	endif	P	1	*Main*	*dot*

2.6 Read Arcs

Table 4: Read Arcs Table

SOURCE		TARGET		MUL.	INSCRIPTION SET	INSCRIPTION
NAME	TYPE	NAME	TYPE			
A	P	_T_6_then	T	1	*Main*	*T*
A	P	_T_7_else	T	1	*Main*	*F*
A	P	else	T	1	*Main*	*x*

Continued on next page

Table 4: Read Arcs Table

SOURCE		TARGET		MUL.	INSCRIPTION SET	INSCRIPTION
NAME	TYPE	NAME	TYPE			
A	P	then	T	1	*Main*	*x*
B	P	_T_0_	T	1	*Main*	*y*
B	P	_T_12_	T	1	*Main*	*y*
B	P	else	T	1	*Main*	*y*
B	P	then	T	1	*Main*	*y*
C	P	_T_12_	T	1	*Main*	*z*

2.7 Inhibitor Arcs

No elements available for description

2.8 Reset Arcs

No elements available for description

2.9 Equal Arcs

No elements available for description

2.10 Comments

No elements available for description

2.11 Images

No elements available for description

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

Table 5: Constants Table

NAME	TYPE	VALUE
T	*bool*	$true$
F	*bool*	$false$

3.2 Functions

No elements available for description

3.3 Simple Color Sets

Table 6: Simple Color Sets Table

NAME	TYPE	COLORS	PLACE COLOR
Dot	*dot*	*dot*	$rgb(217, 191, 255)$
Boolean	*bool*	$true, false$	$rgb(217, 191, 255)$

3.4 Compound Color Sets

No elements available for description

3.5 Alias Color Sets

No elements available for description

3.6 Variables

Table 7: Variables Table

NAME	COLOR SET
x	*Boolean*
y	*Boolean*
z	*Boolean*

4 Hierarchy

This section contains information about the net hierarchy.

4.1 Hierarchy Tree

This section contains the hierarchical tree structure for selected levels.

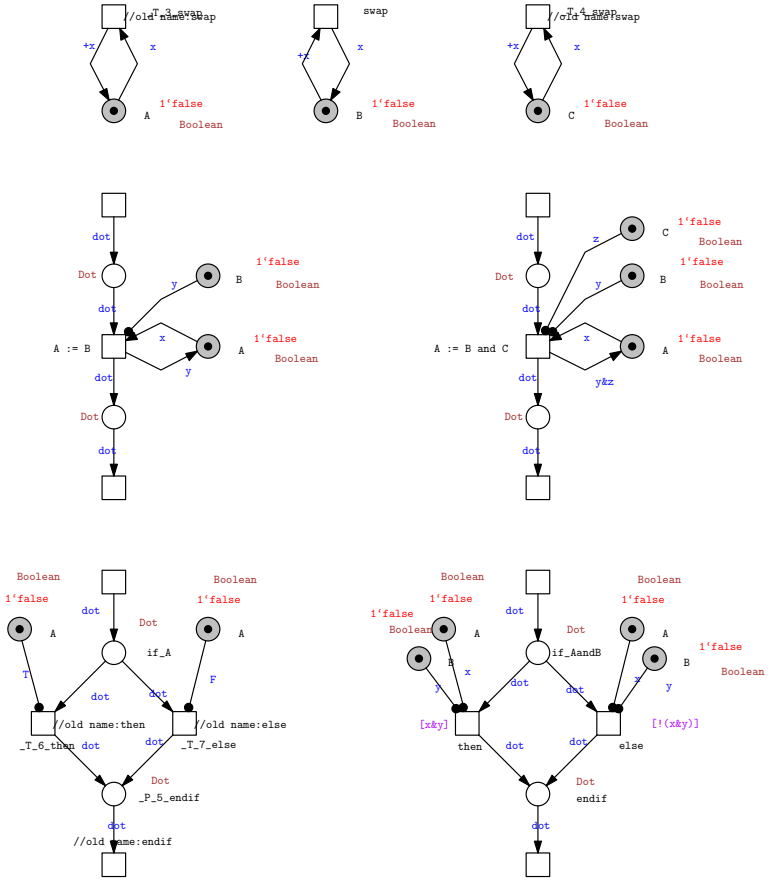
0. Top Level

0.TopLevel

4.2 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.4 Quantitative Nets (Colored)

A.7.4.1 Repressilator.colstochpn (Colored Stochastic Petri Net)

This is a single level net. The top level for this file is shown in Figure A.51. Customizations done in Snoopy2LaTeX dialog:

- Among Graph Elements, only **Place**, **Transition** and **Edge** selected for export
- **Hierarchy** deselected for export

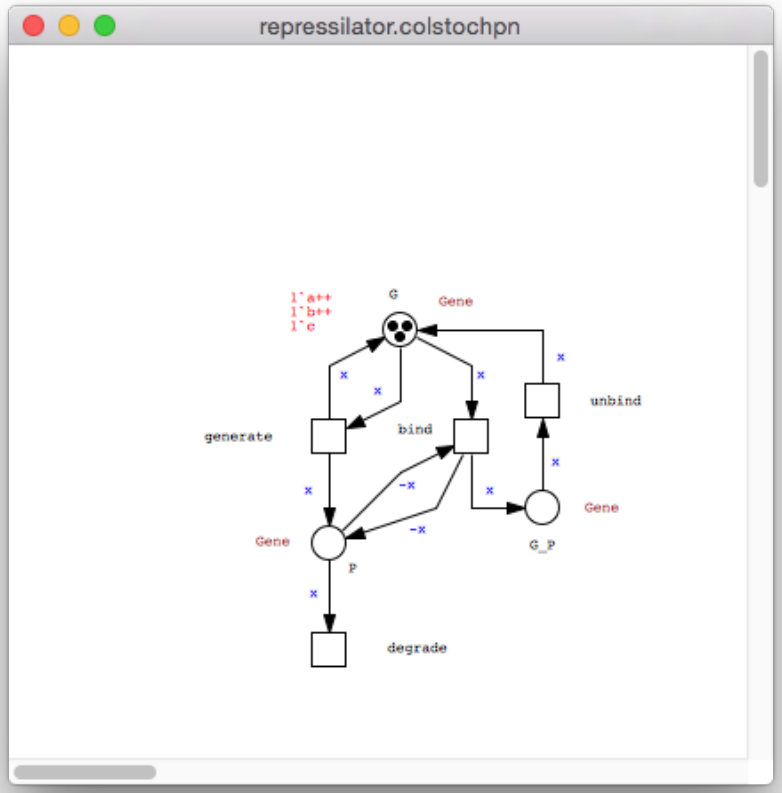
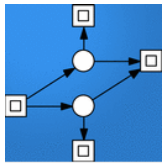


Figure A.51: Repressilator - Snoopy file

The following pages show the generated PDF report.



repressilator.colstochpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

repressilator.colstochpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 18:52

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: repressilator.colstochpn
Creation Timestamp: 2010-02-13 20:13:35

1.2 Net Informations

Net Class: Colored Stochastic Petri Net

Element	Count
Place	3
Transition	4
Immediate Transition	0
Deterministic Transition	0
Scheduled Transition	0
LookupTable	0
Parameter	0
Arc	10
Read Arc	0
Inhibitor Arc	0
Reset Arc	0
Equal Arc	0
Modifier Arc	0

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Places

Table 1: Places Table

NAME	ID	MAR.	LOG.	MAIN: COLOR(S)	MARKING ; M-SET 2: COLOR(S) ; M-SET 2: MARKING	COLORSET	CROSS REFS.
G_P	0	0	✗		; ;	Gene	TopLevel
P	1	0	✗		; ;	Gene	TopLevel
G	2	3	✗	a b c	1; a; 1 1; b; 1 1; c; 1	Gene	TopLevel

2.2 Transitions

Table 2: Transitions Table

NAME	ID	LOG.	PREDICATE	MAIN FUNCTION	GUARD SET	GUARD	CROSS REFS.
bind	0	✗	$true$	$1.0 * P$	$Main$		TopLevel
degrade	1	✗	$true$	$0.001 * P$	$Main$		TopLevel
unbind	2	✗	$true$	$0.0001 * G\_P$	$Main$		TopLevel
generate	3	✗	$true$	$0.1 * G$	$Main$		TopLevel

2.3 Arcs

Table 3: Arcs Table

SOURCE		TARGET		MUL.	EXPRESSION SET	EXPRESSION
NAME	TYPE	NAME	TYPE			
G	P	bind	T	1	$Main$	x
G	P	generate	T	1	$Main$	x
G_P	P	unbind	T	1	$Main$ $unblock\_b$ b $unblock\_c$ c	x 1 1 1 1
P	P	bind	T	1	$Main$	$-x$
P	P	degrade	T	1	$Main$	x
bind	T	G_P	P	1	$Main$	x
bind	T	P	P	1	$Main$	$-x$
generate	T	G	P	1	$Main$	x
generate	T	P	P	1	$Main$	x
unbind	T	G	P	1	$Main$ $unblock\_b$ b $unblock\_c$ c	x 1 1 1 1

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

No elements available for description

3.2 Functions

No elements available for description

3.3 Simple Color Sets

Table 4: Simple Color Sets Table

NAME	TYPE	COLORS	PLACE COLOR
Gene	*enum*	$a - c$	

3.4 Compound Color Sets

No elements available for description

3.5 Alias Color Sets

No elements available for description

3.6 Variables

Table 5: Variables Table

NAME	COLOR SET
x	*Gene*

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.4.2 Volterra.colcontped (Colored Continuous Petri Net)

This is a single level net. The top level for this file is shown in Figure A.52. No customizations are done in Snoopy2LaTeX dialog. Therefore, the report is generated based on the default settings in the dialog.

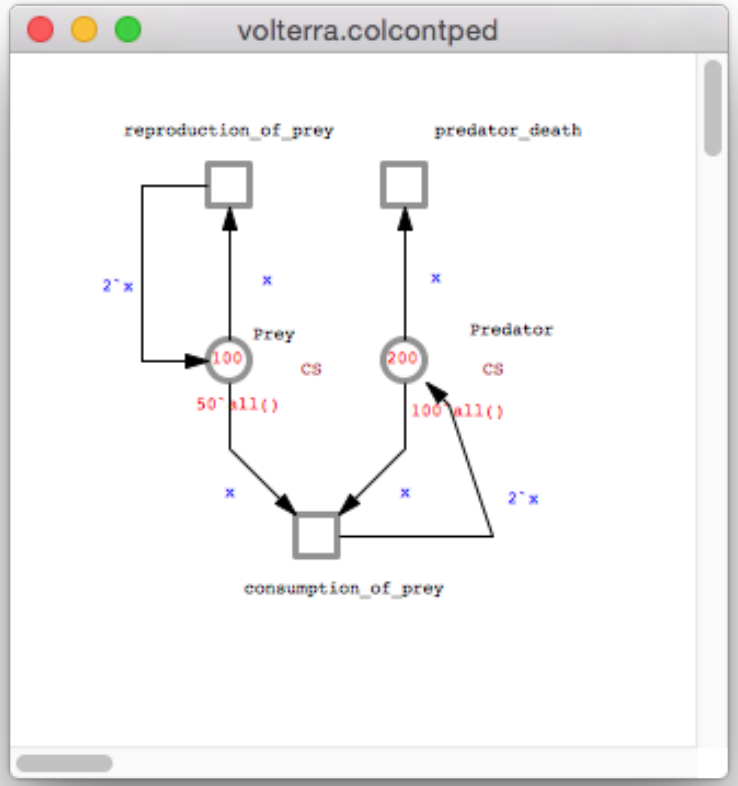
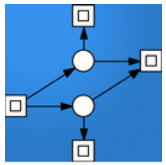


Figure A.52: Volterra - Snoopy file

The following pages show the generated PDF report.



volterra.colcontped

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

volterra.colcontped Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 19:17

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: volterra.colcontped
Creation Timestamp: 2010-09-20 09:41:19

1.2 Net Informations

Net Class: Colored Continuous Petri Net

Element	Count
Continuous Place	2
Continuous Transition	3
Parameter	0
Arc	6
Inhibitor Arc	0
Test Arc	0
Modifier Arc	0

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Continuous Places

Table 1: Continuous Places Table

NAME	ID	MAR.	FIXED	LOG.	MAIN: COLOR(S)	MAIN: MARKING ; M-SET 2: COLOR(S) ; M-SET 2: MARKING	COLORSET	CROSS REFS.
Prey	0	100	0	✗	*all*()	50; *all*(); 100	CS	TopLevel
Predator	1	200	0	✗	*all*()	100; *all*(); 200	CS	TopLevel

2.2 Continuous Transitions

Table 2: Continuous Transitions Table

NAME	PREDICATE	MAIN: FUNCTION ; F-SET 2: FUNCTION	ID	LOG.	REVERSIBLE	GUARD SET	GUARD	CROSS REFS.
reproduction_of_prey	*true*	*MassAction*(1); *Prey*	0	✗	0	*Main*		TopLevel

Continued on next page

Table 2: Continuous Transitions Table

NAME	PREDICATE	MAIN: FUNCTION ; F-SET 2: FUNCTION	ID	LOG.	REVERSIBLE	GUARD SET	GUARD	CROSS REFS.
consumption_of_prey	*true*	*MassAction*(0.005); $0.005 * Prey * Predator$	1	✗	0	*Main*		TopLevel
predator_death	*true*	*MassAction*(0.6); $0.6 * Predator$	2	✗	0	*Main*		TopLevel

2.3 Coarse Places

No elements available for description

2.4 Coarse Transitions

No elements available for description

2.5 Parameters

No elements available for description

2.6 Coarse Parameters

No elements available for description

2.7 Arcs

Table 3: Arcs Table

SOURCE		TARGET		MUL.	EXPRESSION SET	EXPRESSION
NAME	TYPE	NAME	TYPE			
Predator	PC	consumption_of_prey	TC	1	*Main*	x
Predator	PC	predator_death	TC	1	*Main*	x
Prey	PC	consumption_of_prey	TC	1	*Main*	x
Prey	PC	reproduction_of_prey	TC	1	*Main*	x
consumption_of_prey	TC	Predator	PC	1	*Main*	2‘x
reproduction_of_prey	TC	Prey	PC	1	*Main*	2‘x

2.8 Inhibitor Arcs

No elements available for description

2.9 Test Arcs

No elements available for description

2.10 Modifier Arcs

No elements available for description

2.11 Comments

No elements available for description

2.12 Images

No elements available for description

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

No elements available for description

3.2 Functions

No elements available for description

3.3 Simple Color Sets

Table 4: Simple Color Sets Table

NAME	TYPE	COLORS	PLACE COLOR
Dot	*dot*	*dot*	
CS	*enum*	$a - b$	

3.4 Compound Color Sets

No elements available for description

3.5 Alias Color Sets

No elements available for description

3.6 Variables

Table 5: Variables Table

NAME	COLOR SET
x	*CS*

4 Hierarchy

This section contains information about the net hierarchy.

4.1 Hierarchy Tree

This section contains the hierarchical tree structure for selected levels.

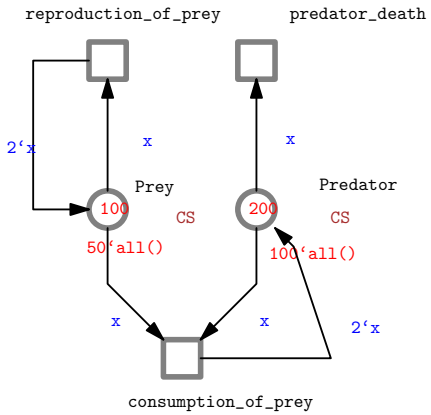
0. Top Level

0.TopLevel

4.2 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

PC Place, Continuous

TC Transition, Continuous

A.7.4.3 2D_diffusion_hpnc.colhybpn (Colored Hybrid Petri Net)

This is a single level net. The top level for this file is shown in Figure A.53. Customizations done in Snoopy2LaTeX dialog:

- Among Transition types, all others except **Stochastic Transition** and **Continuous Transition** deselected for export
- **Parameter** and **Coarse Parameter** deselected for export
- **General Informations** deselected for export
- 'Generate Hierarchy Tree' deselected for export

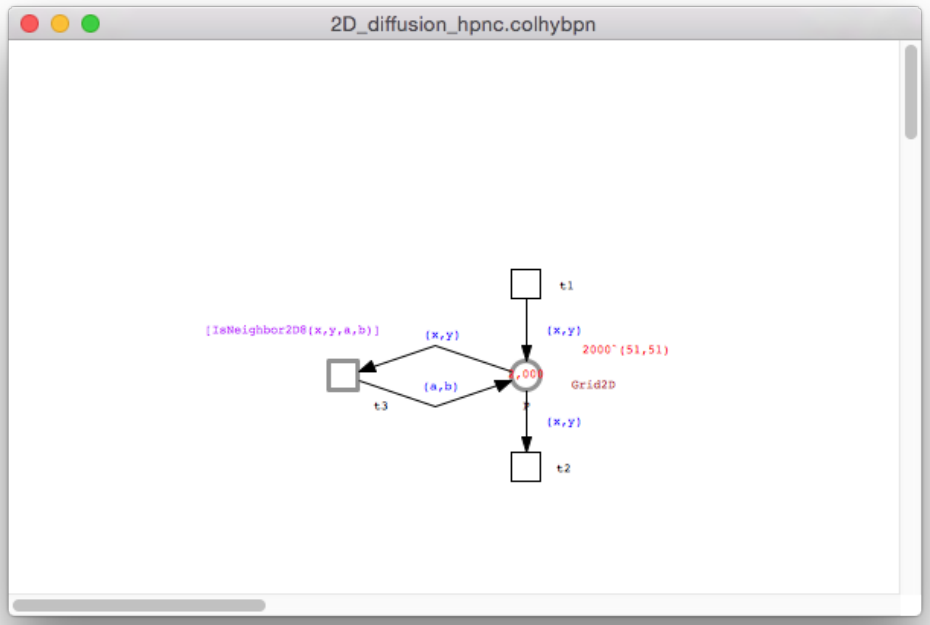
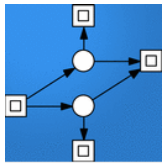


Figure A.53: Reaction diffusion systems - Snoopy file

The following pages show the generated PDF report.



2D_diffusion_hpnc.colhybpn

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

2D_diffusion_hpnc.colhybpn Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 20:10

Contents

1 Basics

This section contains basic information about the input net.

1.1 Net Informations

Net Class: Colored Hybrid Petri Net

Element	Count
Discrete Place	0
Continuous Place	1
Stochastic Transition	2
Continuous Transition	1
Immediate Transition	0
Deterministic Transition	0
Scheduled Transition	0
Parameter	0
Arc	4
Read Arc	0
Inhibitor Arc	0
Reset Arc	0
Equal Arc	0
Modifier Edge	0

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Discrete Places

No elements available for description

2.2 Continuous Places

Table 1: Continuous Places Table

FIXED	NAME	ID	MAR.	LOG.	MAIN: COLOR(S)	MAIN: MARKING	COLORSET	CROSS REFS.
0	P	0	2000	✗	$(51, 51)$	2000	Grid2D	TopLevel

2.3 Stochastic Transitions

Table 2: Stochastic Transitions Table

NAME	ID	LOG.	PREDICATE	MAIN: FUNCTION	GUARD SET	GUARD	CROSS REFS.
t1	0	✗	*true*	$MassAction(0.0015)$	*Main*		TopLevel
t2	1	✗	*true*	$MassAction(0.001)$	*Main*		TopLevel

2.4 Continuous Transitions

Table 3: Continuous Transitions Table

NAME	ID	LOG.	PREDICATE	MAIN: FUNCTION	GUARD SET	GUARD	CROSS REFS.
t3	2	✗	*true*	$MassAction(0.1)$	*Main*	$IsNeighbor2D8(x, y, a, b)$	TopLevel

2.5 Coarse Places

No elements available for description

2.6 Arcs

Table 4: Arcs Table

SOURCE		TARGET		MUL.	EXPRESSION SET	EXPRESSION
NAME	TYPE	NAME	TYPE			
P	P	t2	T	1	*Main*	(x, y)
P	P	t3	T	1	*Main*	(x, y)
t1	T	P	P	1	*Main*	(x, y)
t3	T	P	P	1	*Main*	(a, b)

2.7 Read Arcs

No elements available for description

2.8 Inhibitor Arcs

No elements available for description

2.9 Reset Arcs

No elements available for description

2.10 Equal Arcs

No elements available for description

2.11 Modifier Edges

No elements available for description

2.12 Comments

No elements available for description

2.13 Images

No elements available for description

3 Declarations

This section contains information related to declarations specific to the net.

3.1 Constants

Table 5: Constants Table

NAME	TYPE	VALUE
$D1$	*int*	101
$D2$	*int*	101

3.2 Functions

Table 6: Functions Table

RETURN TYPE	FUNCTION NAME	PARAMETER LIST	FUNCTION BODY
bool	$IsNeighbor2D8$	$CD1x, CD2y, CD1a, CD2b$	$(a = x\|a = x + 1\|a = x - 1)\&(b = y\|b = y+1\|b = y-1)\&(!(a = x\&b = y))\&(a <= D1\&b <= D2)\&(a >= 1\&b >= 1)$

3.3 Simple Color Sets

Table 7: Simple Color Sets Table

NAME	TYPE	COLORS	PLACE COLOR
Dot	*dot*	*dot*	*white*
$CD1$	*int*	$1 - D1$	*white*
$CD2$	*int*	$1 - D2$	*white*

3.4 Compound Color Sets

Table 8: Compound Color Sets Table

NAME	TYPE	COMPONENT COLOR SETS	COLORS	PLACE COLOR
Grid2D	*product*	$CD1, CD2$		*white*

3.5 Alias Color Sets

No elements available for description

3.6 Variables

Table 9: Variables Table

NAME	COLOR SET
x	$CD1$
y	$CD2$
a	$CD1$
b	$CD2$

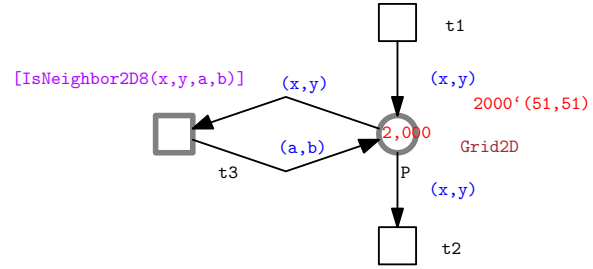
4 Hierarchy

This section contains information about the net hierarchy.

4.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place, Continuous

T Transition

A.7.5 Other Petri Nets

A.7.5.1 LinearLogic.spfreen (Freestyle Net)

This is a single level net. The top level for this file is shown in Figure A.54. Customizations done in Snoopy2LaTeX dialog:

- Among Graph Elements, only **Circle**, **Edge** and **Comment** selected for export
- **Declarations** deselected for export
- 'Generate Hierarchy Tree' deselected for export

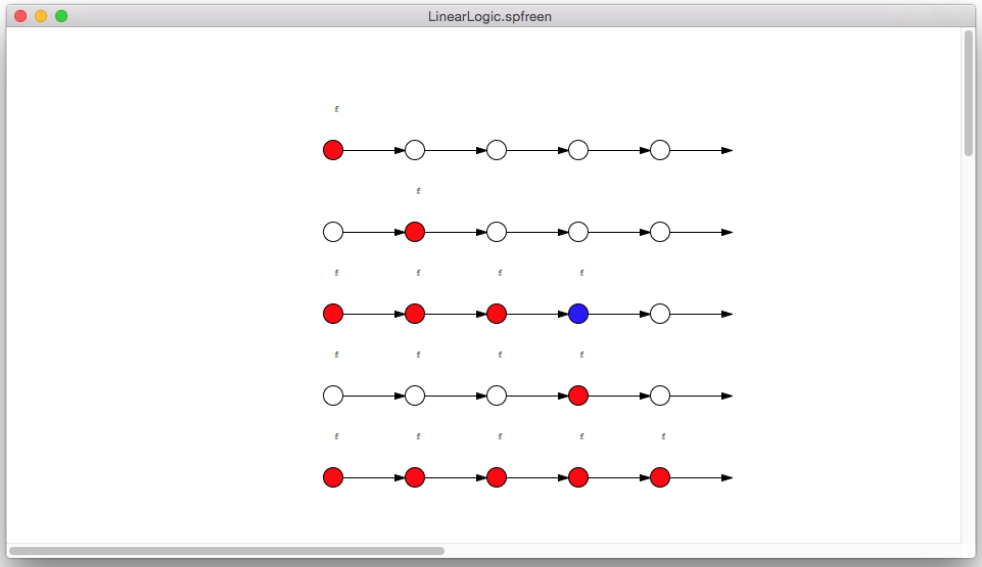
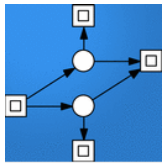


Figure A.54: LinearLogic - Snoopy file

The following pages show the generated PDF report.



LinearLogic.spfreen

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

17/06/2015

LinearLogic.spfreen Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

17/06/2015 19:25

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: LinearLogic.spfreen
Creation Timestamp: 2011-07-20 14:45:15

1.2 Net Informations

Net Class: Freestyle Net

Element	Count
Edge	25

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Circles

Table 1: Circles Table

NAME	CROSS REFS.
_C_300_	TopLevel
_C_306_	TopLevel
_C_312_	TopLevel
_C_318_	TopLevel
_C_324_	TopLevel
_C_330_	TopLevel
_C_487_	TopLevel
_C_493_	TopLevel
_C_499_	TopLevel
_C_505_	TopLevel
_C_511_	TopLevel
_C_517_	TopLevel
_C_643_	TopLevel
_C_649_	TopLevel
_C_655_	TopLevel
_C_661_	TopLevel
_C_667_	TopLevel
_C_673_	TopLevel
_C_799_	TopLevel
_C_805_	TopLevel
_C_811_	TopLevel
_C_817_	TopLevel
_C_823_	TopLevel
_C_829_	TopLevel
_C_955_	TopLevel
_C_961_	TopLevel
_C_967_	TopLevel
_C_973_	TopLevel
_C_979_	TopLevel
_C_985_	TopLevel

2.2 Edges

Table 2: Edges Table

SOURCE		TARGET			
NAME	TYPE	NAME	TYPE	NAME	SPLINE
_C_300_	C	_C_306_	C		1
_C_306_	C	_C_312_	C		1
_C_312_	C	_C_318_	C		1
_C_318_	C	_C_324_	C		1
_C_324_	C	_C_330_	C		1
_C_493_	C	_C_487_	C		1
_C_499_	C	_C_493_	C		1
_C_505_	C	_C_499_	C		1
_C_511_	C	_C_505_	C		1
_C_517_	C	_C_511_	C		1
_C_643_	C	_C_649_	C		1
_C_649_	C	_C_655_	C		1
_C_655_	C	_C_661_	C		1
_C_661_	C	_C_667_	C		1
_C_667_	C	_C_673_	C		1
_C_805_	C	_C_799_	C		1
_C_811_	C	_C_805_	C		1
_C_817_	C	_C_811_	C		1
_C_823_	C	_C_817_	C		1
_C_829_	C	_C_823_	C		1
_C_955_	C	_C_961_	C		1
_C_961_	C	_C_967_	C		1
_C_967_	C	_C_973_	C		1
_C_973_	C	_C_979_	C		1
_C_979_	C	_C_985_	C		1

2.3 Comments

Table 3: Comments Table

NET REFERENCE	COMMENT
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f

Continued on next page

Table 3: Comments Table

NET REFERENCE	COMMENT
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f
TopLevel	f

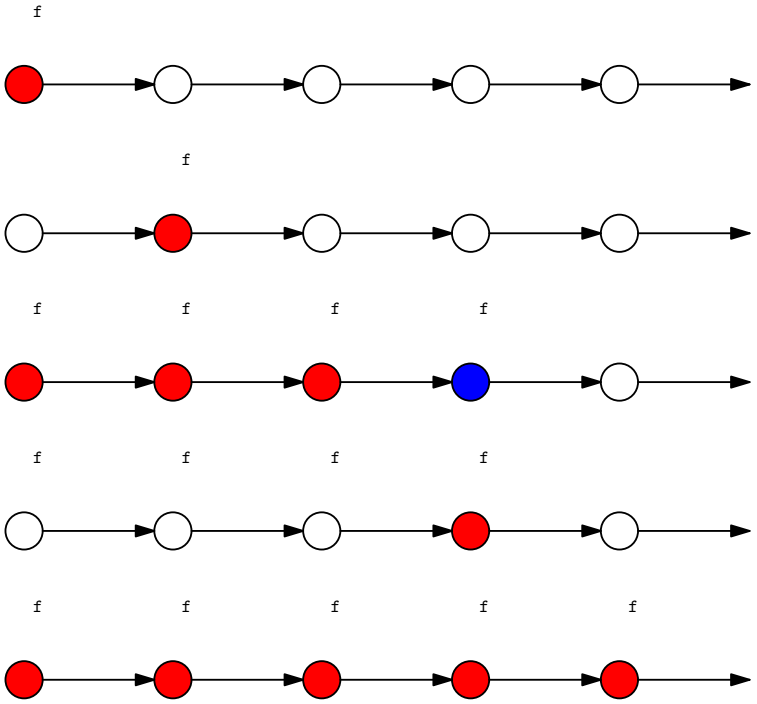
3 Hierarchy

This section contains information about the net hierarchy.

3.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

C Circle

A.7.5.2 Modulo_demo.spmnet (Modulo Petri Net)

This is a single level net. The top level for this file is shown in Figure A.55. Customizations done in Snoopy2LaTeX dialog:

- Among Graph Elements, **Coarse Place** and **Coarse Transition** deselected for export
- **Comments** reordered to precede all other Graph Elements.
- **Declarations** deselected for export
- 'Generate Hierarchy Tree' deselected for export

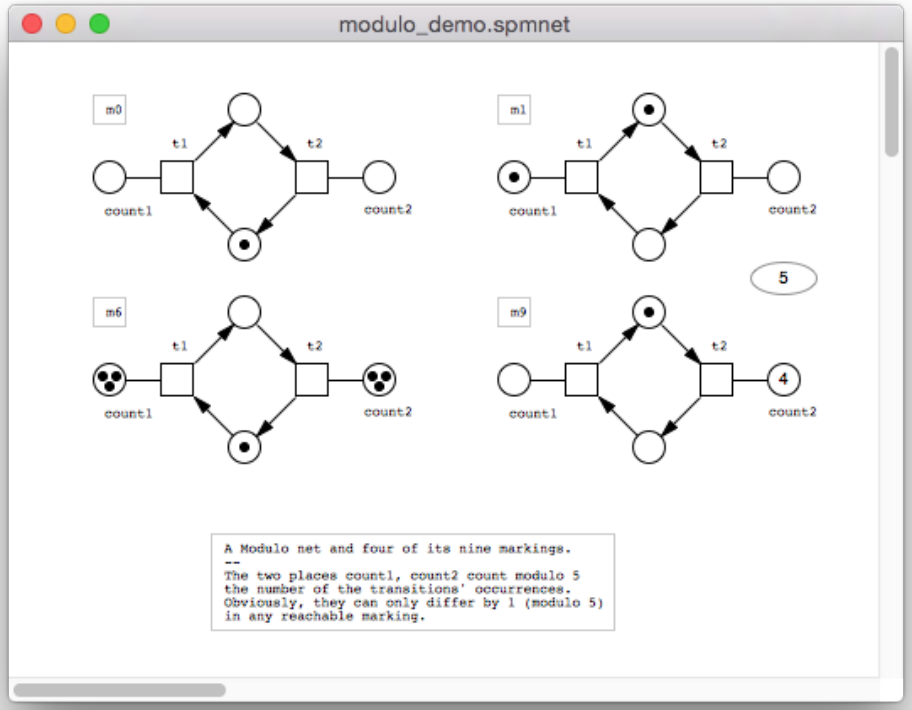
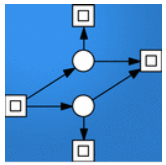


Figure A.55: Modulo_demo - Snoopy file

The following pages show the generated PDF report.



modulo_demo.spmnet

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

18/06/2015

modulo_demo.spmnet Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

18/06/2015 16:14

Contents

1 Basics

This section contains basic information about the input net.

1.1 General Informations

File Name: modulo_demo.spmnet
Creation Timestamp: 2015-06-18 16:08:39

1.2 Net Informations

Net Class: Modulo Petri Net

Element	Count
Place	16
Transition	8
Arc	16
Undirected Edge	8

2 Graph Elements

This section contains information related to graph elements specific to the net.

2.1 Comments

Table 1: Comments Table

NET REFERENCE	COMMENT
TopLevel	m0
TopLevel	A Modulo net and four of its nine markings. – The two places count1, count2 count modulo 5 the number of the transitions' occurrences. Obviously, they can only differ by 1 (modulo 5) in any reachable marking.
TopLevel	m1
TopLevel	m6
TopLevel	m9

2.2 Places

Table 2: Places Table

NAME	ID	MAR.	LOG.	CROSS REFS.
_P_0_	0	0	✗	TopLevel
_P_1_	1	1	✗	TopLevel
_P_2_count1	2	1	✗	TopLevel
_P_3_count2	3	0	✗	TopLevel
_P_4_count2	4	4	✗	TopLevel
_P_5_count1	5	0	✗	TopLevel
_P_6_	6	1	✗	TopLevel
_P_7_	7	0	✗	TopLevel
_P_8_	8	1	✗	TopLevel
_P_9_	9	0	✗	TopLevel
_P_10_count1	10	0	✗	TopLevel
_P_11_count2	11	0	✗	TopLevel
_P_12_	12	1	✗	TopLevel
_P_13_	13	0	✗	TopLevel
count1	14	3	✗	TopLevel
count2	15	3	✗	TopLevel

2.3 Transitions

Table 3: Transitions Table

NAME	ID	LOG.	CROSS REFS.
_T_0_t1	0	✗	TopLevel
_T_1_t2	1	✗	TopLevel
_T_2_t2	2	✗	TopLevel
_T_3_t1	3	✗	TopLevel
_T_4_t1	4	✗	TopLevel
_T_5_t2	5	✗	TopLevel
t1	6	✗	TopLevel
t2	7	✗	TopLevel

2.4 Modulos

Table 4: Modulos Table

UID	MODULO	CROSS REFS.
M9930	5	TopLevel

2.5 Arcs

Table 5: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
_P_0_	P	_T_0_t1	T	1
_P_12_	P	t1	T	1
_P_13_	P	t2	T	1
_P_1_	P	_T_1_t2	T	1
_P_6_	P	_T_2_t2	T	1
_P_7_	P	_T_3_t1	T	1
_P_8_	P	_T_4_t1	T	1
_P_9_	P	_T_5_t2	T	1
_T_0_t1	T	_P_1_	P	1
_T_1_t2	T	_P_0_	P	1
_T_2_t2	T	_P_7_	P	1
_T_3_t1	T	_P_6_	P	1

Continued on next page

Table 5: Arcs Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
_T_4_t1	T	_P_9_	P	1
_T_5_t2	T	_P_8_	P	1
t1	T	_P_13_	P	1
t2	T	_P_12_	P	1

2.6 Undirected Edges

Table 6: Undirected Edges Table

SOURCE		TARGET		
NAME	TYPE	NAME	TYPE	MUL.
_P_10_count1	P	_T_4_t1	T	1
_P_11_count2	P	_T_5_t2	T	1
_P_2_count1	P	_T_0_t1	T	1
_P_3_count2	P	_T_1_t2	T	1
_P_4_count2	P	_T_2_t2	T	1
_P_5_count1	P	_T_3_t1	T	1
count1	P	t1	T	1
count2	P	t2	T	1

2.7 Images

No elements available for description

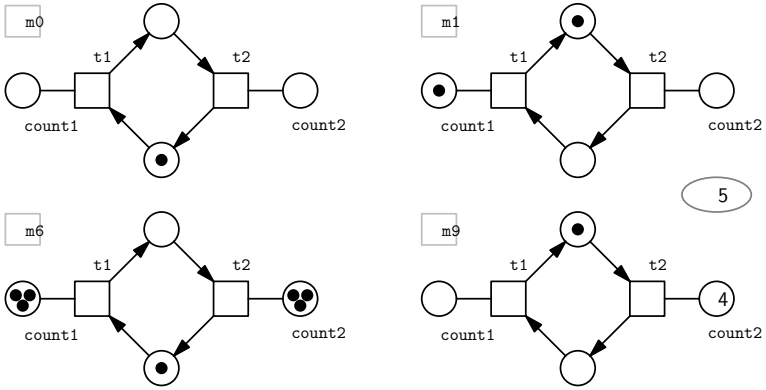
3 Hierarchy

This section contains information about the net hierarchy.

3.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



A Modulo net and four of its nine markings.
--
The two places count1, count2 count modulo 5
the number of the transitions' occurrences.
Obviously, they can only differ by 1 (modulo 5)
in any reachable marking.

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

P Place

T Transition

A.7.5.3 Bdgraph.spmtbdd (MTBDD)

This is a single level net. The top level for this file is shown in Figure A.56. Customizations done in Snoopy2LaTeX dialog:

- **Basics** and **Declarations** deselected for export
- 'Generate Hierarchy Tree' deselected for export

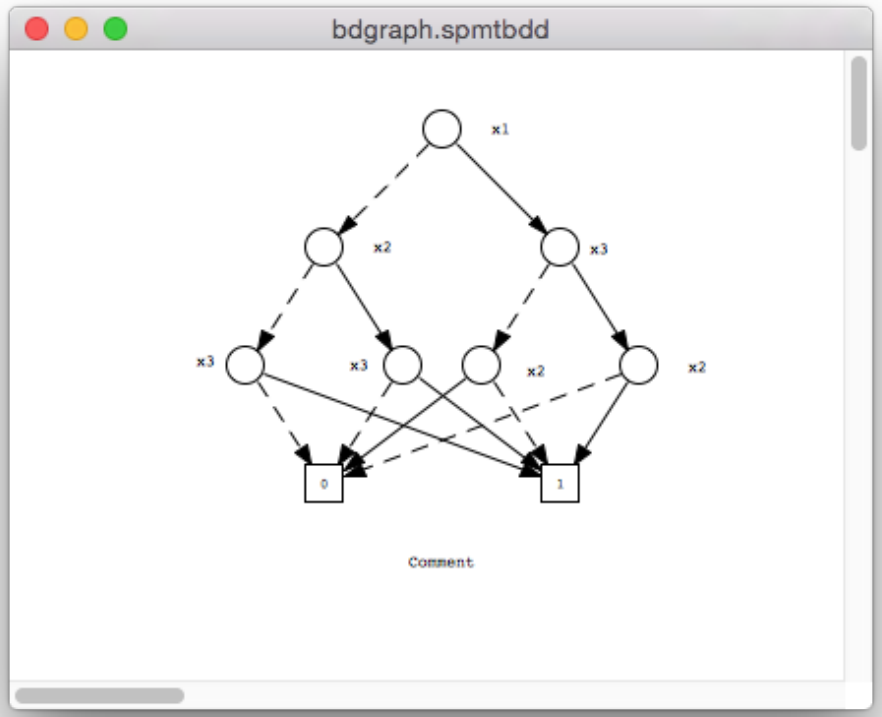
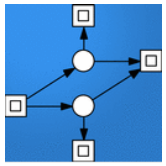


Figure A.56: Bdgraph - Snoopy file

The following pages show the generated PDF report.



bdgraph.spmtbdd

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

18/06/2015

bdgraph.spmtbdd Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

18/06/2015 16:40

Contents

1 Graph Elements

This section contains information related to graph elements specific to the net.

1.1 Inner Nodes

Table 1: Inner Nodes Table

UID	VARIABLE	ISROOT	CROSS REFS.
I136	x1	0	TopLevel
I144	x2	0	TopLevel
I152	x3	0	TopLevel
I160	x3	0	TopLevel
I168	x3	0	TopLevel
I184	x2	0	TopLevel
I192	x2	0	TopLevel

1.2 Terminal Nodes

Table 2: Terminal Nodes Table

UID	VALUE	CROSS REFS.
T205	0	TopLevel
T208	1	TopLevel

1.3 Highs

Table 3: Highs Table

SOURCE		TARGET	
UID	TYPE	UID	TYPE
I136	I	I152	I
I144	I	I168	I
I152	I	I184	I
I160	I	T208	T
I168	I	T208	T
I184	I	T208	T
I192	I	T205	T

1.4 Lows

Table 4: Lows Table

SOURCE		TARGET	
UID	TYPE	UID	TYPE
I136	I	I144	I
I144	I	I160	I
I152	I	I192	I
I160	I	T205	T
I168	I	T205	T
I184	I	T205	T
I192	I	T208	T

1.5 Comments

Table 5: Comments Table

NET REFERENCE	COMMENT
TopLevel	Comment

1.6 Images

No elements available for description

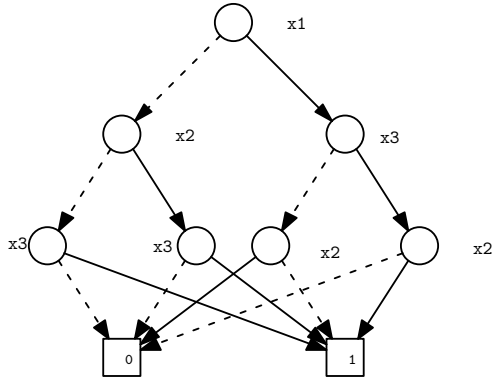
2 Hierarchy

This section contains information about the net hierarchy.

2.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



Comment

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

I Inner Node

T Terminal Node

A.7.5.4 Roidd.spmtidd (MTIDD)

This is a single level net. The top level for this file is shown in Figure A.57. Customizations done in Snoopy2LaTeX dialog:

- The order of all sections reversed
- 'Generate Hierarchy Tree' deselected for export

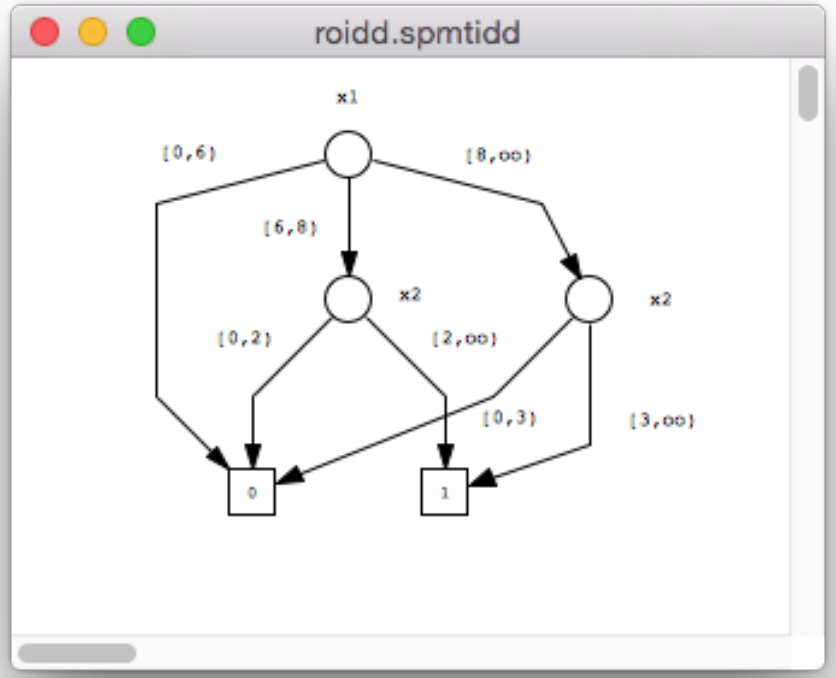
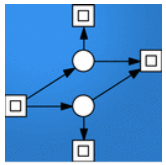


Figure A.57: Roidd - Snoopy file

The following pages show the generated PDF report.



roidd.spmtidd

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy

18/06/2015

roidd.spmtidd Snoopy2LaTeX

This document has been generated with Snoopy [1, 2].
If you have any comments or suggestions,
Please contact snoopy@informatik.tu-cottbus.de

18/06/2015 16:44

Contents

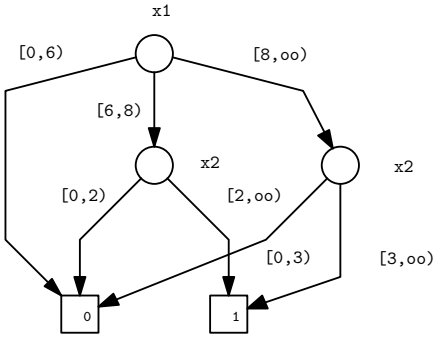
1 Hierarchy

This section contains information about the net hierarchy.

1.1 Hierarchy Figures

This section contains the hierarchical figures for selected levels.

0. Top Level



2 Declarations

This section contains information related to declarations specific to the net.

3 Graph Elements

This section contains information related to graph elements specific to the net.

3.1 Inner Nodes

Table 1: Inner Nodes Table

UID	VARIABLE	ISROOT	CROSS REFS.
I136	x1	0	TopLevel
I152	x2	0	TopLevel
I160	x2	0	TopLevel

3.2 Terminal Nodes

Table 2: Terminal Nodes Table

UID	VALUE	CROSS REFS.
T168	0	TopLevel
T171	1	TopLevel

3.3 Edges

Table 3: Edges Table

SOURCE		TARGET		
UID	TYPE	UID	TYPE	INTERVAL
I136	I	I152	I	6,8
I136	I	I160	I	8,oo
I136	I	T168	T	0,6
I152	I	T168	T	0,2
I152	I	T171	T	2,oo
I160	I	T168	T	0,3
I160	I	T171	T	3,oo

3.4 Comments

No elements available for description

3.5 Images

No elements available for description

4 Basics

This section contains basic information about the input net.

4.1 General Informations

File Name: roidd.spmtidd
Creation Timestamp: 2015-06-18 16:44:15

4.2 Net Informations

Net Class: MTIDD

Element	Count
Edge	7

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick. Snoopy a unifying Petri net tool. Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398-407.

[2] C. Rohr, W. Marwan and M. Heiner. Snoopy - a unifying Petri net framework to investigate biomolecular networks. Bioinformatics, 26(7):974-975, 2010.

Glossary

CROSS REFS. Cross-References

LOG. Logic

MAR. Marking

MUL. Multiplicity

NET. Net number

I Inner Node

T Terminal Node

A.7.5.5 (Fault Tree)

A.7.5.6 (Extended Fault Tree)

A.8 Further Reading

References

[1] M. Heiner, M. Herajy, F. Liu, C. Rohr and M. Schwarick, *Snoopy a unifying Petri net tool*, Proc. PETRI NETS 2012, Volume 7347, Springer, June 2012, pages 398–407.

[2] C. Rohr, W. Marwan and M. Heiner, *Snoopy - a unifying Petri net framework to investigate biomolecular networks*, Bioinformatics, 26(7): 974–975, 2010.

[3] LaTeX - Bibliography Management, `http://en.wikibooks.org/wiki/LaTeX/Bibliography_Management` (Last accessed: April 20, 2015)

[4] Regular Expression, `http://en.wikipedia.org/wiki/Regular_expression` (Last accessed: May 10, 2015)

B Snoopy Graph Classes

According to [1], Snoopy is primarily divided into two levels: *uncolored* and *colored*. Each level comprises a family of related Petri net classes, sharing structure, but being specialized by their kinetic information. Specifically, the uncolored level contains qualitative (time-free) Petri nets (***QPN***) as well as quantitative (time-dependent) Petri nets such as stochastic Petri nets (***SPN***), continuous Petri nets (***CPN***), and generalized hybrid Petri nets (***GHPN***). The colored level consists of the colored counterparts of the uncolored level, thus containing colored qualitative Petri nets ($\boldsymbol{QPN}^C$), colored stochastic Petri nets ($\boldsymbol{SPN}^C$), colored continuous Petri nets ($\boldsymbol{QPN}^C$), and colored generalized hybrid Petri nets ($\boldsymbol{CPN}^C$).

Snoopy supports a large number of graph classes and unique file formats specific to each class. The available graph classes in Snoopy (grouped under related Petri net classes) can be listed as:

- **Uncolored Petri Nets**
  - **Qualitative Petri Nets**
    1. Petri Net (*.pn, *.spped)
    2. Extended Petri Net (*.xpn, *.spept)
    3. Reachability Graph (*.sprgraph)
    4. Music Petri Net (*.spm)
  - **Quantitative Petri Nets**
    1. Stochastic Petri Net (*.spn, *.spstochpn)
    2. Continuous Petri Net (*.cpn, *.spcontped)
    3. Hybrid Petri Net (*.hpn, *.sphybrid)
    4. Time Petri Net (*.tpn, *.sptpt)
- **Colored Petri Nets**
  - **Qualitative Petri Nets (Colored)**
    1. Colored Petri Net (*.colpn)
    2. Colored Extended Petri Net (*.colxpn, *.colextpn)
  - **Quantitative Petri Nets (Colored)**
    1. Colored Stochastic Petri Net (*.colspn, *.colstochpn)
    2. Colored Continuous Petri Net (*.colcpn, *.colcontped)
    3. Colored Hybrid Petri Net (*.colhpn, *.colhybpn)
- **Further Petri Nets**
  1. Fault Tree (*.spft)
  2. Extended Fault Tree (*.speft)
  3. Freestyle Net (*.spfreen)
  4. Modulo Petri Net (*.spmnet)
  5. MTBDD (*.spmtbdd)
  6. MTIDD (*.spmtidd)