Structural model refinement as type refinement of colored Petri nets

Diana-Elena Gratie, Ion Petre

Turku Centre for Computer Science
Åbo Akademi University

BioPPN 2015
Brussels, June 22

Summary

- What, why, how
- Reaction-based models
- Structural data refinement
- Type refinement of colored Petri nets
- Coloring strategy
- Conclusions and future work

What

Structural

data

refinement

of reaction-based models

Why

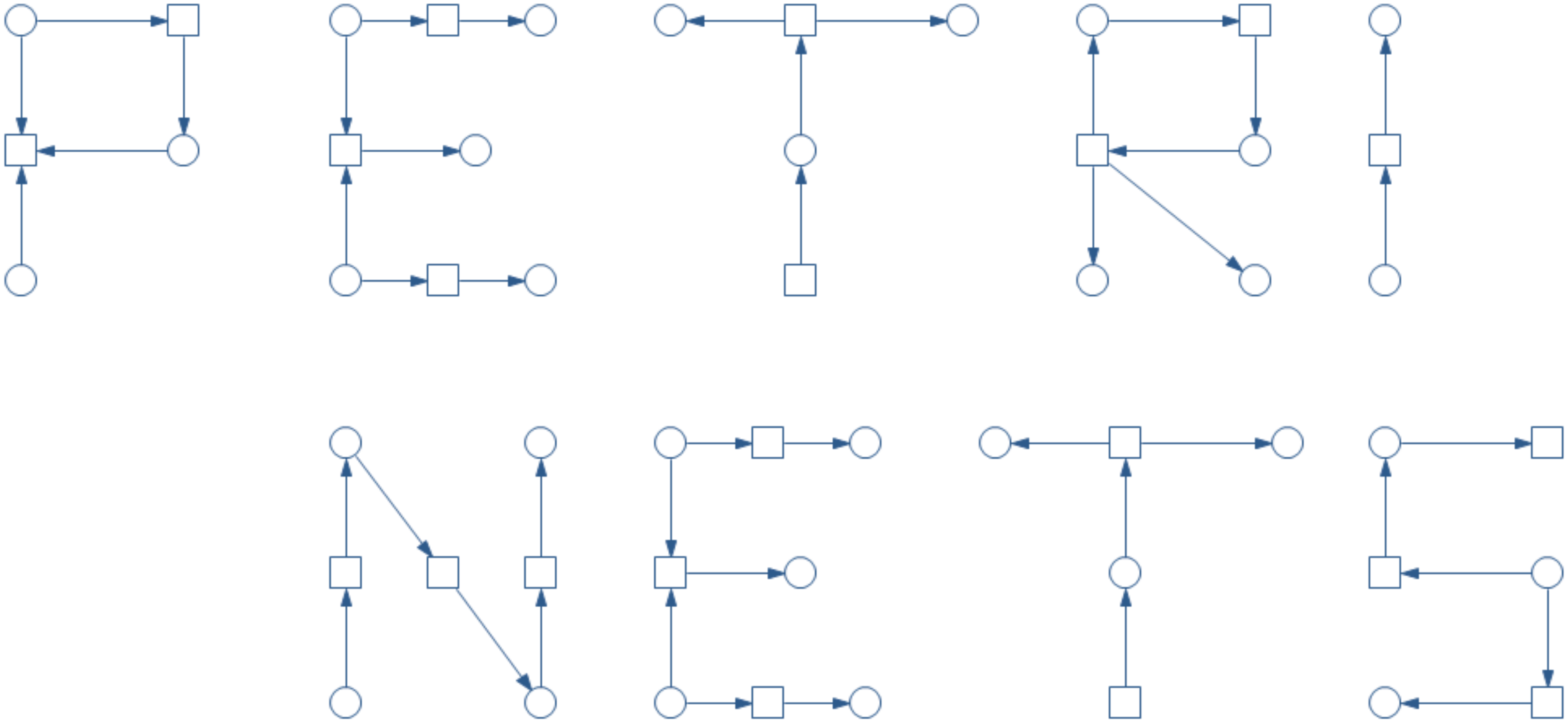
"Building models with different levels of complexity and identifying robust features relevant to the biological problem remains an important research strategy. "

Qiang Cui and Ruth Nussinov, Making Biomolecular Simulations Accessible in the Post-Nobel Prize Era, PLOS Computational Biology

Why

- Model-building is a stepwise process
- As new data becomes available, new models should reflect the new insights
- It is less time-consuming to add new details to existing (ready-fitted) models
- Model refinement could be automated!

How



Reaction-based models

- $r_j : c_{1,j} S_1 + \cdots + c_{m,j} S_m \rightarrow c'_{1,j} S_1 + \cdots + c'_{m,j} S_m$

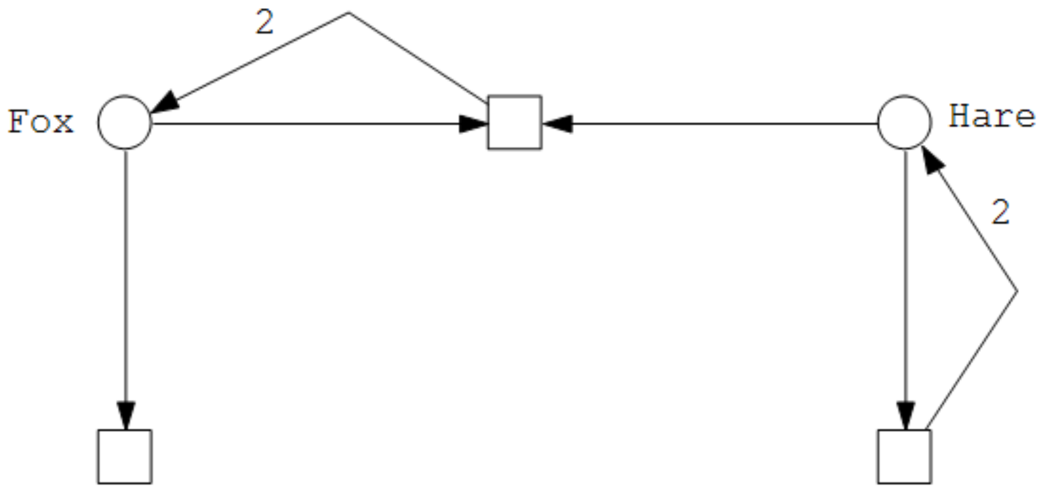
$$2H_2 + O_2 \rightarrow 2H_2O$$

$$2P \rightarrow P_2$$

 +  → 

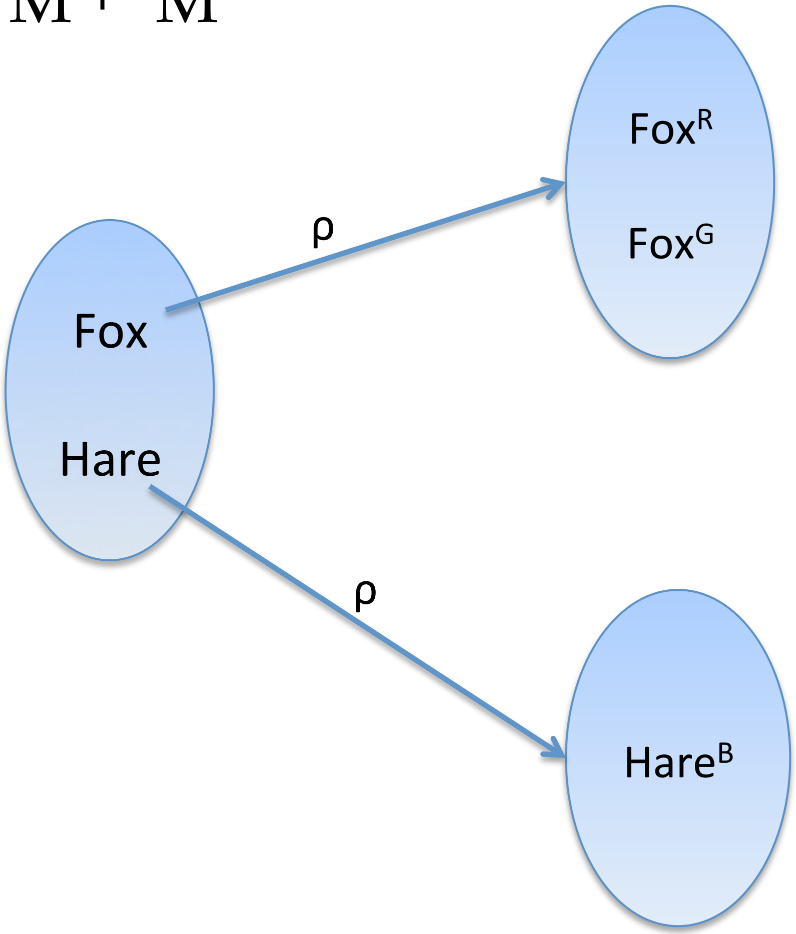
Reaction-based models

- M=(***S, R***)
  - ***S*** **=** {Fox, Hare}
  - ***R*** **=** $\{r\downarrow 1, r\downarrow 2, r\downarrow 3\}$
    - $r\downarrow 1$: Hare → 2Hare
    - $r\downarrow 2$: Fox + Hare → 2Fox
    - $r\downarrow 3$: Fox → Φ



Structural data refinement

- $M \mapsto^{\rho} \dashv M'$



Full structural data refinement

$M = (S, R)$

- S = {Fox, Hare}
- $R = \{r_1, r_2, r_3\}$
  - r_1 : Hare → 2Hare
  - r_2 : Fox + Hare → 2Fox
  - r_3 : Fox → Φ

{(Fox, Fox^R), (Fox, Fox^G)} ⊂ ρ

{(Hare, $Hare^B$)} ⊂ ρ

$\{(r_1, r'_1)\} \subset \rho$

$\{(r_2, r'_2), (r_2, r'_3), (r_2, r'_4), (r_2, r'_5), (r_2, r'_6), (r_2, r'_7)\} \subset \rho$

$\{(r_3, r'_8), (r_3, r'_9)\} \subset \rho$

$M_\rho = (S', R')$

- S' = {Fox^R, Fox^G, $Hare^B$}
- $R' = \{r'_1, r'_2, r'_3, r'_4, r'_5, r'_6, r'_7, r'_8, r'_9\}$
  - r'_1 : $Hare^B$ → $2Hare^B$
  - r'_2 : Fox^R + $Hare^B$ → $2Fox^R$
  - r'_3 : Fox^R + $Hare^B$ → Fox^R+Fox^G
  - r'_4 : Fox^R + $Hare^B$ → $2Fox^G$
  - r'_5 : Fox^G + $Hare^B$ → $2Fox^R$
  - r'_6 : $FoxG$ + $Hare^B$ → Fox^R+Fox^G
  - r'_7 : Fox^G + $Hare^B$ → $2Fox^G$
  - r'_8 : Fox^R → Φ
  - r'_9 : Fox^G → Φ

Type refinement of colored Petri nets

- Charles Lakos. ***Composing abstractions of coloured petri nets***. In *Application and Theory of Petri Nets 2000, pages 323–342. Springer, 2000*

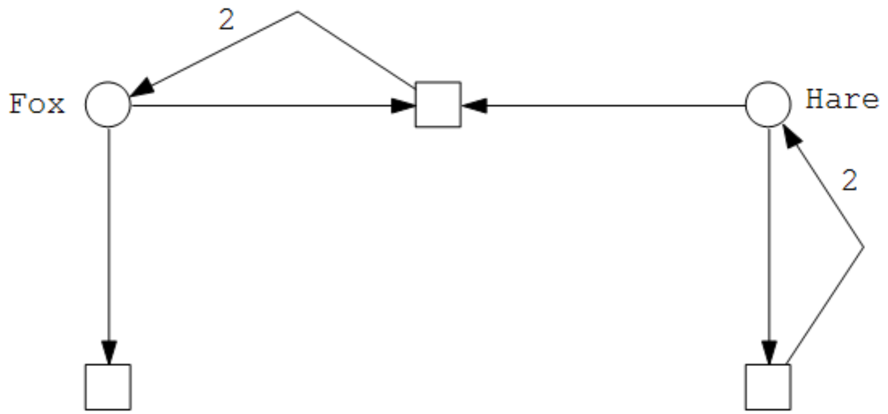
- Altering the color sets of places in a colored Petri net N such that the new color sets are polymorphic with the old ones respectively yields a colored Petri net N' such that there is a type refinement morphism between them.

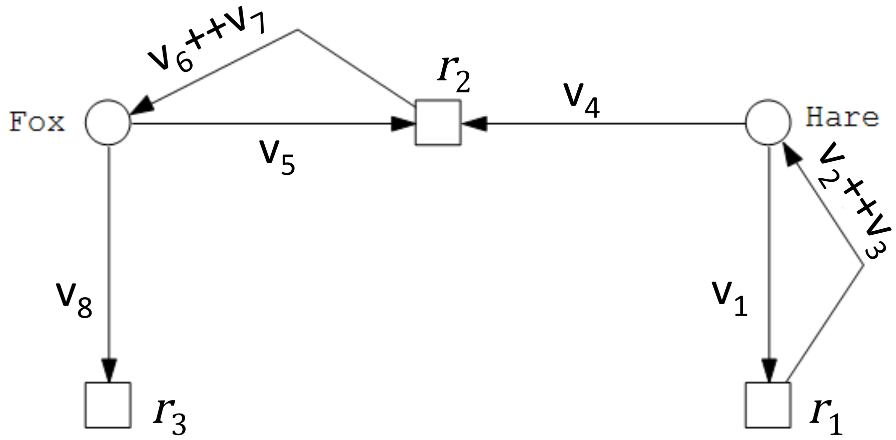
- A type refinement morphism is a behavior-respecting mapping of two colored Petri nets.

Coloring strategy

- Given a standard Petri net $N=(P,T,A,f,M_0)$:
  - *For places, use as color sets records with a fixed number of fields.*
  - *For transitions, use multisets over the color sets of pre- and post- places, with multiplicities given by arc expressions*

Coloring strategy example





color sets:

- enum name_f := fox
- enum name_h := hare
- record foxcol with name:name_f
- record harecol with name:name_h
- multiset hares := harecol, harecol, harecol
- multiset foxeshare := foxcol, foxcol, foxcol, harecol
- multiset foxdeath := foxcol

variables:

- harecol v_1, v_2, v_3, v_4
- foxcol v_5, v_6, v_7, v_8

C(Fox) = foxcol

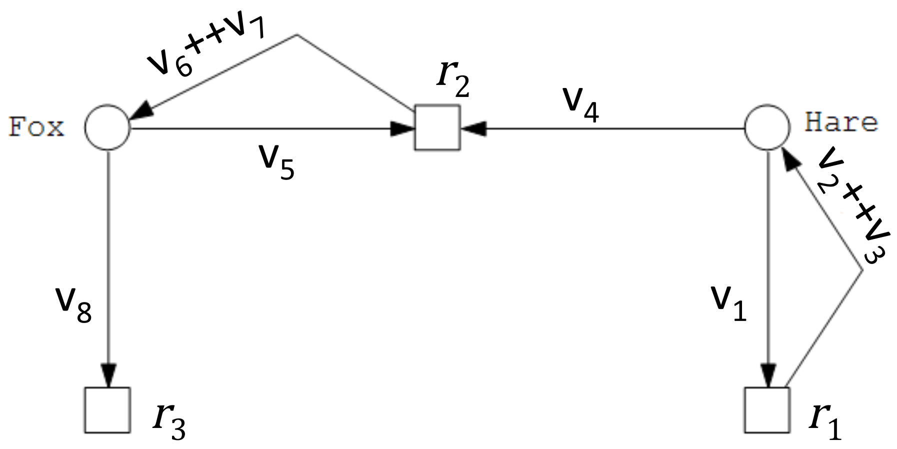
C(Hare) = harecol

$C(r_1)$ = hares

$C(r_2)$ = foxeshare

$C(r_3)$ = foxdeath

Coloring strategy example



variables:

harecol v_1, v_2, v_3, v_4

foxcol v_5, v_6, v_7, v_8

C(Fox) = foxcol

C(Hare) = harecol

C(r_1) = hares

C(r_2) = foxeshare

C(r_3) = foxdeath

color sets:

enum name_f := fox

enum name_h := hare

record foxcol with name:name_f

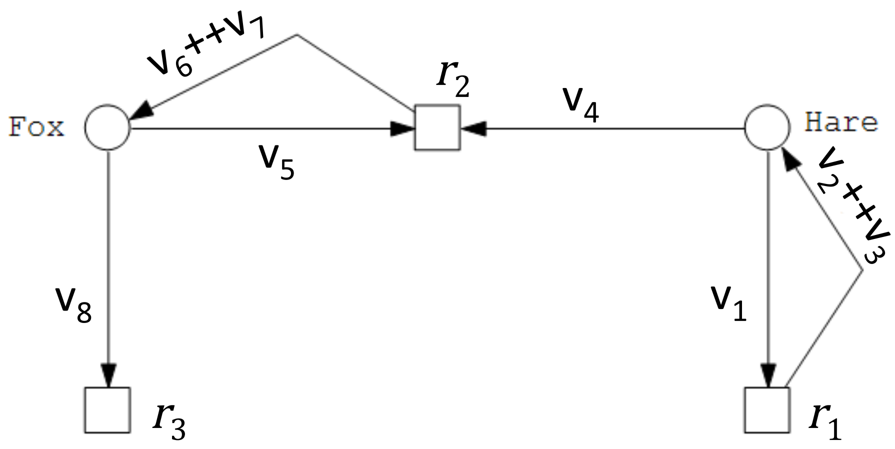
record harecol with name:name_h

multiset hares := harecol, harecol, harecol

multiset foxeshare := foxcol, foxcol, foxcol, harecol

multiset foxdeath := foxcol

Coloring strategy example - refinement



color sets:

enum name_f := fox

enum name_h := hare

enum fox_fur := red, gray

enum hare_fur := brown

record foxcol’ with name:name_f,
    fur:fox_fur

record harecol’ with name:name_h
    fur:hare_fur

multiset hares’ := harecol’, harecol’,
    harecol’

multiset foxeshare’ := foxcol’,
    foxcol’, foxcol’, harecol’

multiset foxdeath’ := foxcol’

variables:

harecol v_1, v_2, v_3, v_4

foxcol v_5, v_6, v_7, v_8

C(Fox) = foxcol

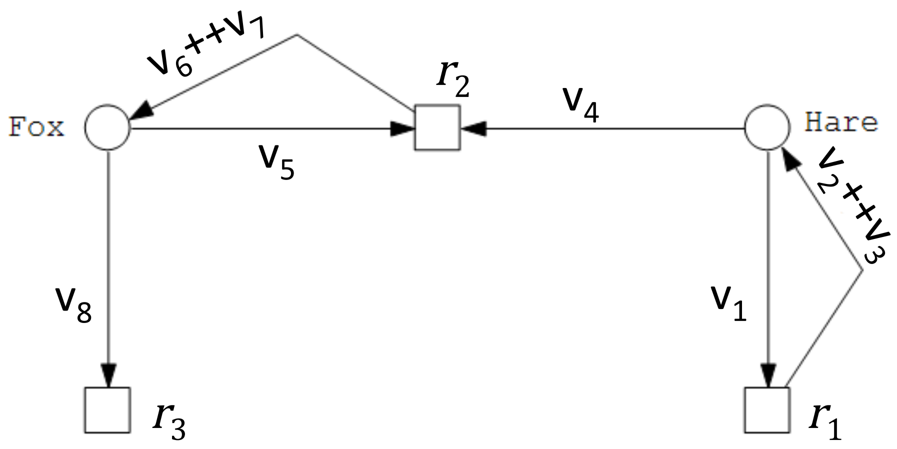
C(Hare) = harecol

C(r_1) = hares

C(r_2) = foxeshare

C(r_3) = foxdeath

Coloring strategy example - refinement



variables:

harecol' v_1, v_2, v_3, v_4

foxcol' v_5, v_6, v_7, v_8

C(Fox) = foxcol'

C(Hare) = harecol'

C(r_1) = hares'

C(r_2) = foxeshare'

C(r_3) = foxdeath'

color sets:

enum name_f := fox

enum name_h := hare

enum fox_fur := red, gray

enum hare_fur := brown

record foxcol' with name:name_f,
fur:fox_fur

record harecol' with name:name_h
fur:hare_fur

multiset hares' := harecol', harecol',
harecol'

multiset foxeshare' := foxcol',
foxcol', foxcol', harecol'

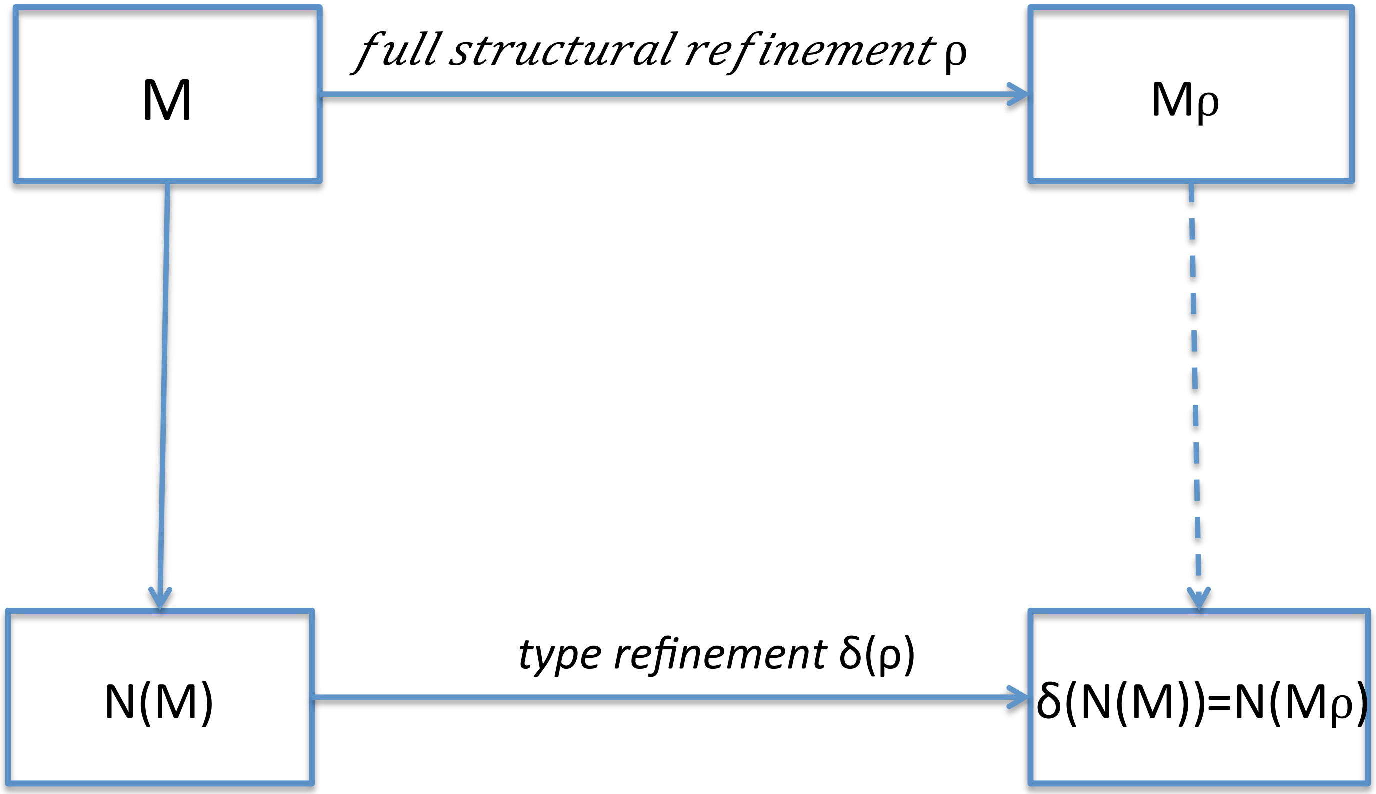
multiset foxdeath' := foxcol'

Refinement algorithm

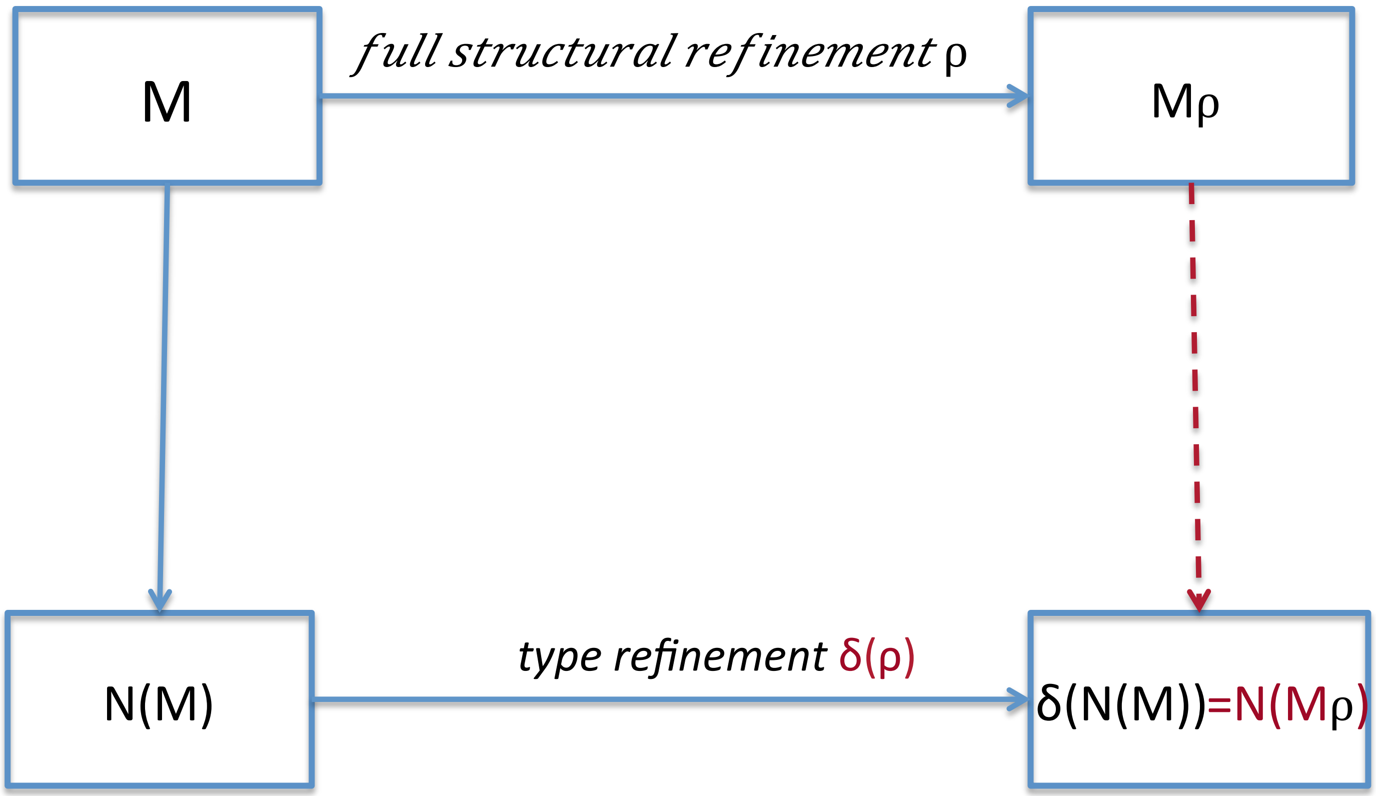
Algorithm 1 TypeRef

function TypeRef(N, ρ)
  $\Sigma' \leftarrow \varnothing$;
  ▷ create the new color sets based on the old ones;
  for all $p \in P$ **do**
    $cs \leftarrow C(p)$;
    define a new color set cs' that extends cs with a new field with $\rho(\delta^{-1}(p))$ values;
    $\Sigma' \leftarrow \Sigma' \cup \{cs'\}$;
    $C'(p) \leftarrow cs'$;
  end for
  for all $t \in T$ **do**
    define cs as a multiset $cs : \{C'(p) \mid p \in P\} \rightarrow \mathbb{N}$ such that $cs(C'(p)) = C(t)(C(p)),\ \forall p \in P$;
    $\Sigma' \leftarrow \Sigma' \cup \{cs\}$;
    $C'(t) \leftarrow cs$;
  end for
  ▷ re-type the arc expressions: for each variable in an arc expression, create one having as type the new color set of the place that the arc is connected to; the new arc expression is a multiset sum of these variables;
  $E' \leftarrow \varnothing$;
  for all $e \in E$ **do**
    $p \leftarrow$ the place connected to e;
    $V \leftarrow$ set of variables appearing in e;
    $V' \leftarrow \varnothing$;
    for all $v_i \in V$ **do**
      define $v'_i : C'(p)$;
      $V' \leftarrow V' \cup \{v'_i\}$
    end for
    $e' \leftarrow {}^{++}\sum_{v \in V'} v$; ▷ ${}^{++}\sum$ denotes multiset addition;
    $E' \leftarrow E' \cup \{e'\}$;
  end for
  $\mathbb{M}' \leftarrow \mu\{(p, c) \mid p \in P, c \in C'(p)\}$;
  $\mathbb{Y}' \leftarrow \mu\{(t, c) \mid t \in T, c \in C'(t)\}$;
  $\mathbb{M}'_0$ is designed such that $\sum_{c \in C'(p)} \mid \mathbb{M}'_0(p, c) \mid = \mid \mathbb{M}_0(p, \mathcal{C}(p)) \mid, \forall p \in P$;
  $N' \leftarrow (P, T, A, \Sigma', C', E', \mathbb{M}', \mathbb{Y}', M'_0)$;
return N';
end function

Conclusions and future work



Conclusions and future work



Conclusions and future work

- We provide a flexible approach for building step-by-step larger models, while:
  - preserving the network structure
  - adding details in the color sets

- Next stage:
  - use this technique to refine a large biomodel
  - study the scalability of this technique
  - find the sweet spot

Thank you!

dgratie@abo.fi