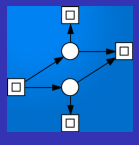
Spatial Encoding of Systems Using Coloured Petri Nets

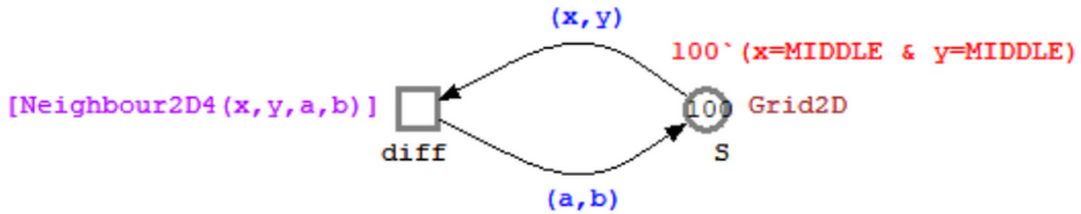
George Assaf and Monika Heiner

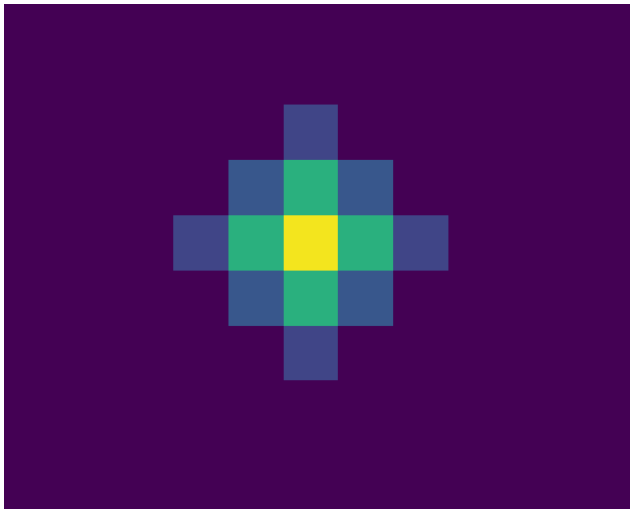
Computer Science Institute, Faculty 1 Brandenburg University of Technology, Cottbus, Germany

Hagen, 10-11 October, 2019

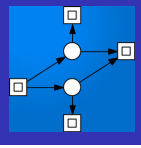
Outline





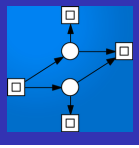


The Problem

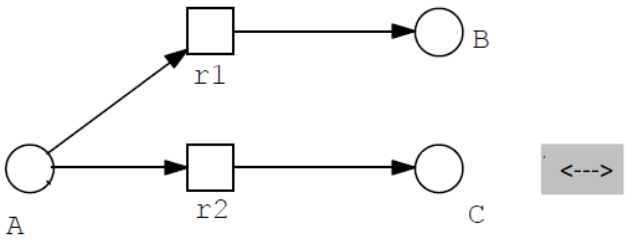


- The explicit notion of space is a crucial point for many systems.
  - Biochemical diffusion which develops over space.
  - A moving Robot which occupies a certain size in space.
- Modelling such systems requires taking location information into consideration
  - e.g, in biochemical diffusion, diffusion can only be happened between two neighbouring-grid positions.
- Colored Petri nets ($\mathcal{PN}^{\mathcal{C}}$) are an excellent formalism for modeling and analysing such systems.
- We introduce two space-encoding principles using coloured Petri nets to address this issue.

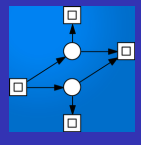
Coloured Petri nets



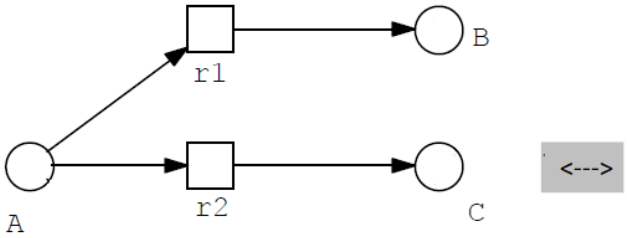
- No colours.


B
r1
A
r2
C
<--->

Coloured Petri nets

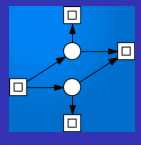


- Coloured version.

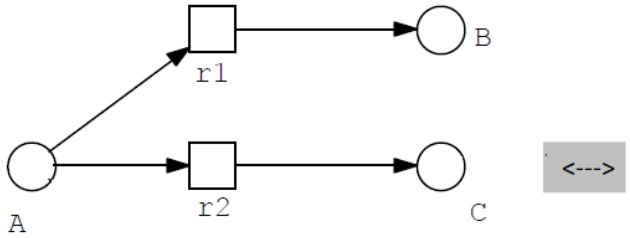



```
Declarations:
colorset Dot= dot;
colorset CS= enum with B, C;
var      x : CS;
```

Coloured Petri nets

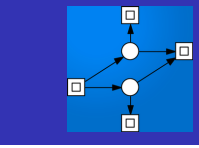


- Coloure definitions.

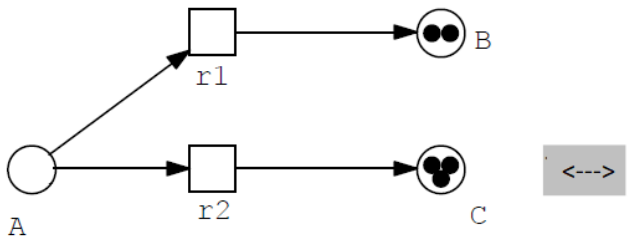



```
Declarations:
colorset Dot= dot;
colorset CS= enum with B, C;
var      x : CS;
```

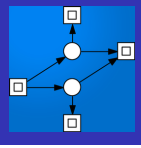
Coloured Petri nets



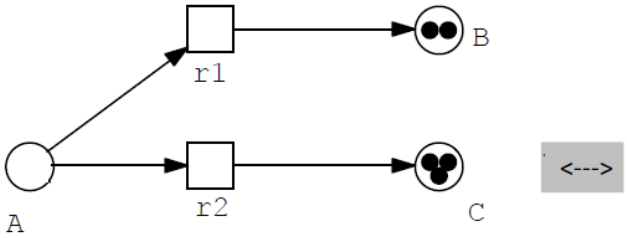
- With tokens.



Coloured Petri nets

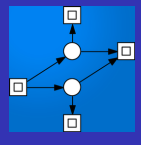


- With tokens.

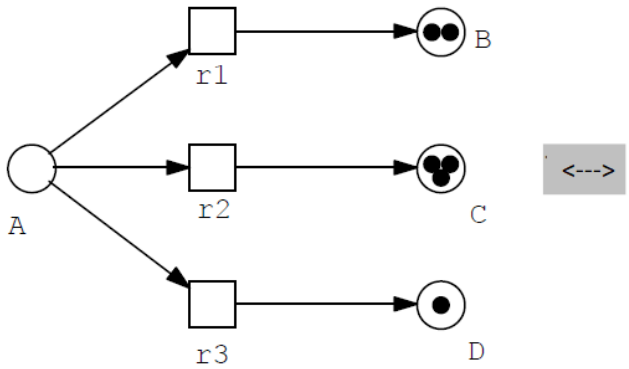



```
Declarations:
colorset Dot= dot;
colorset CS= enum with B, C;
var      x : CS;
```

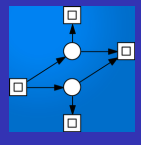
Coloured Petri nets



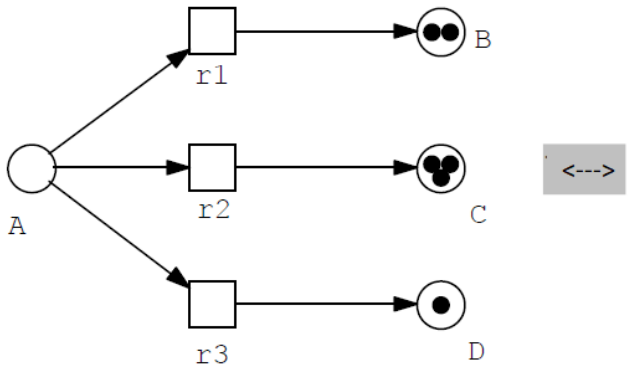
- More complex structure.



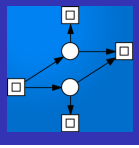
Coloured Petri nets



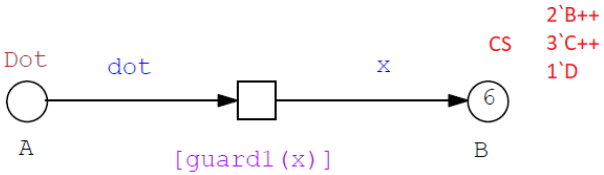
- Adjust colour definitions.



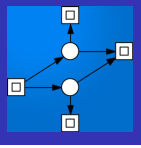
Coloured Petri nets



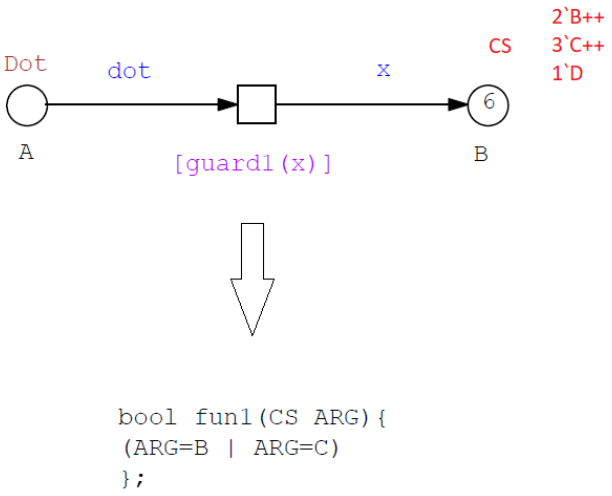
- with Guard.



Coloured Petri nets

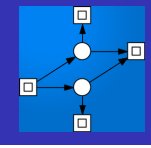


- with Guard.

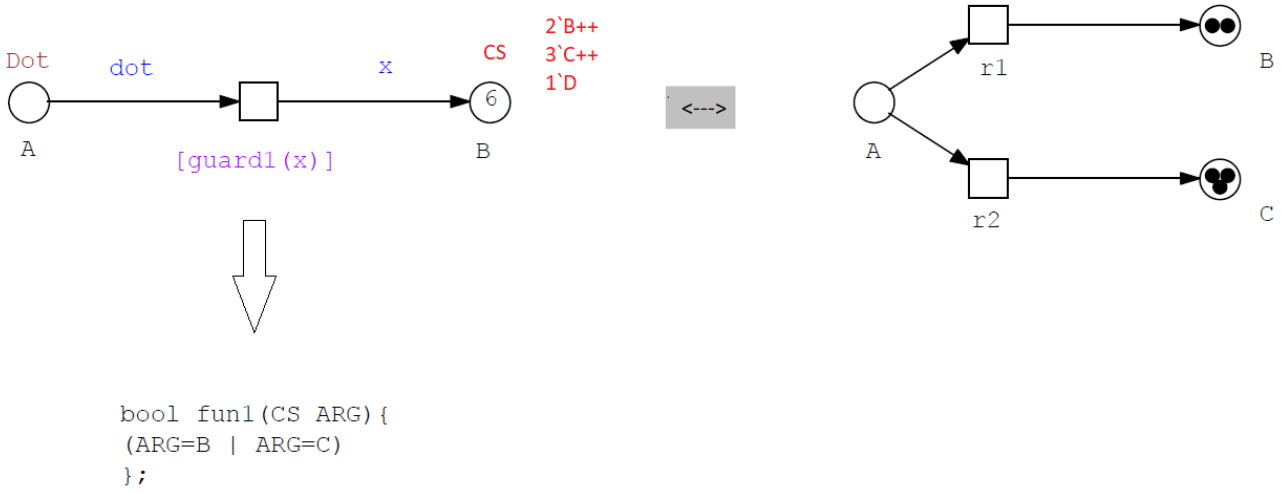



```
bool fun1(CS ARG){
(ARG=B | ARG=C)
};
```

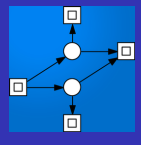
Coloured Petri nets



- with Guard.

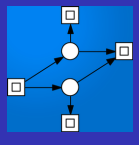


Coloured quantitative Petri nets

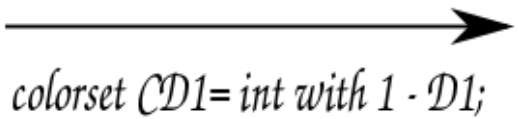


- Coloured stochastic Petri nets ($\mathcal{SPN}^{\mathcal{C}}$)
  - discrete number of tokens on its places -> non-negative integer.
  - stochastic firing rate is associated with each transition.
- Coloured continuous Petri nets ($\mathcal{CPN}^{\mathcal{C}}$)
  - continuous amount of tokens on its places -> real value.
  - deterministic firing rate is associated with each transition.
- Both stochastic and continuous Petri nets transitions follow firing rates specified by arbitrary mathematical function.

First spatial encoding scheme

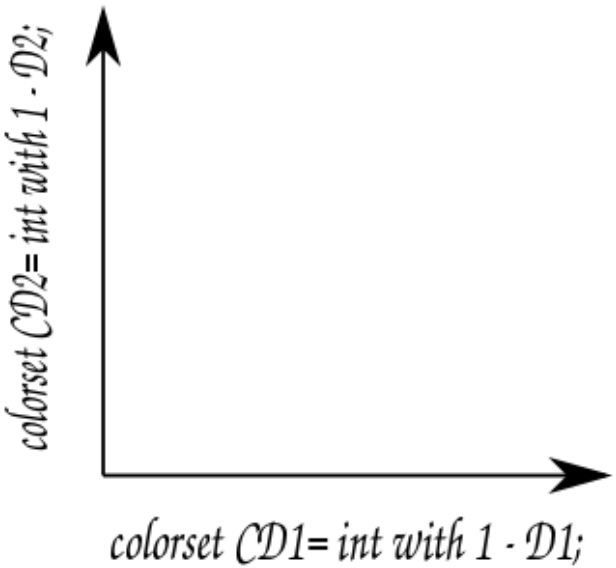


- First Dimension.



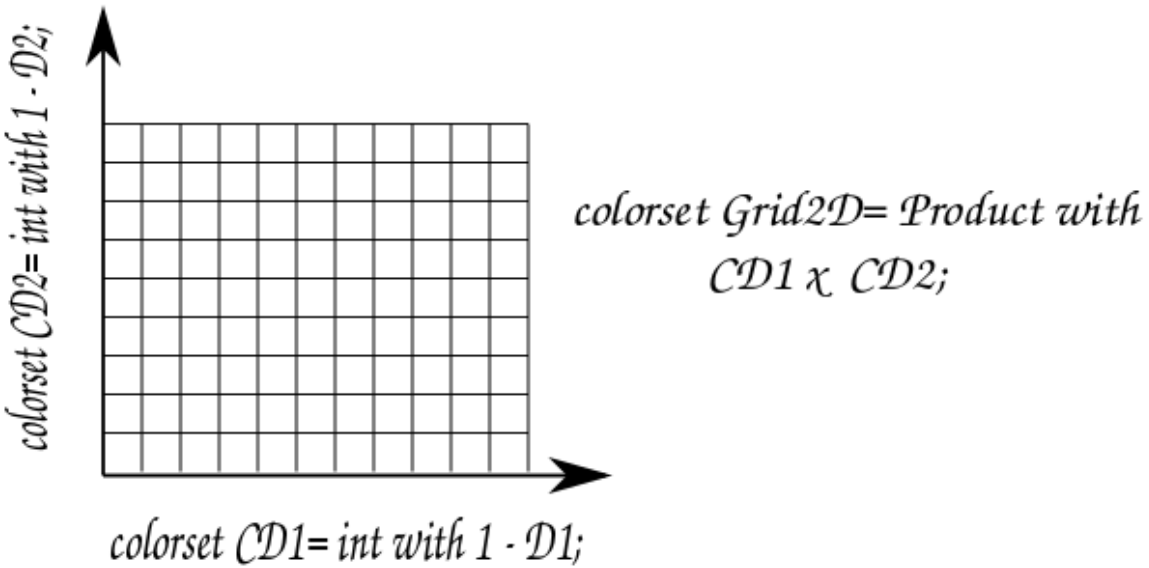
First spatial encoding scheme

- Second Dimension.

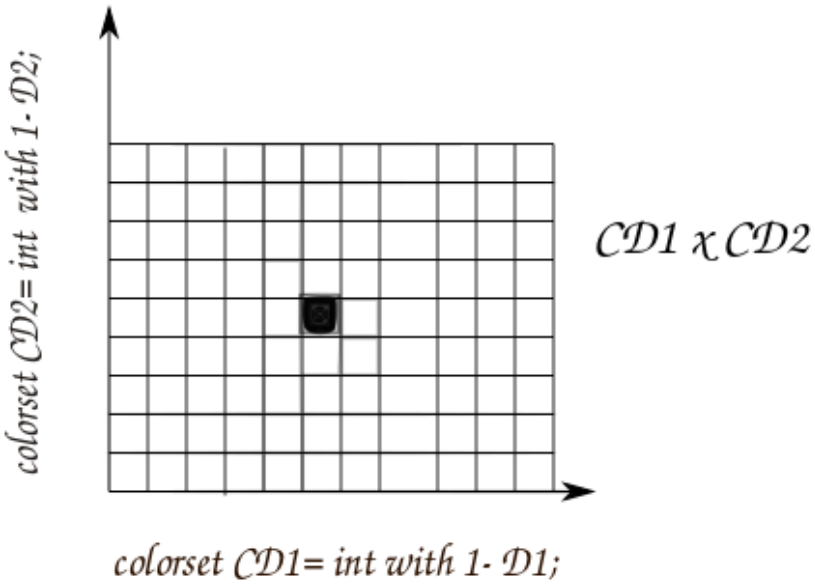


First spatial encoding scheme

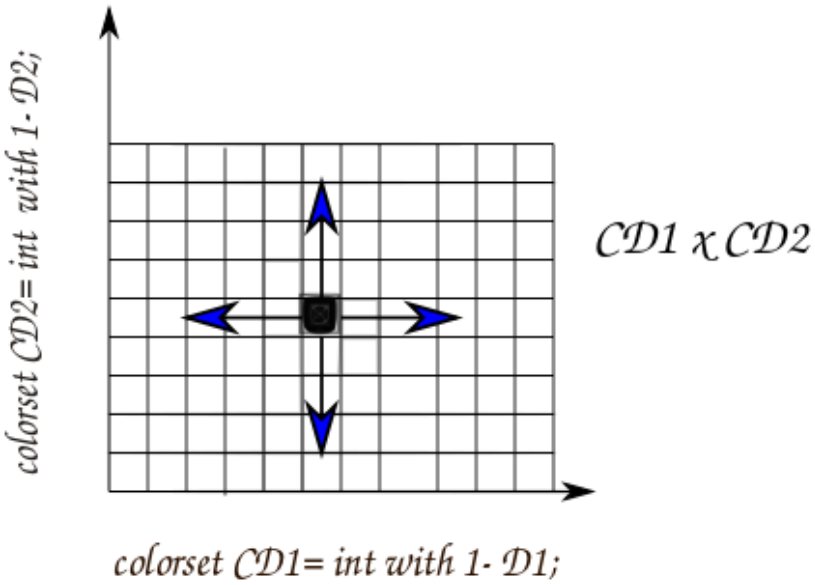
- Grid 2D.



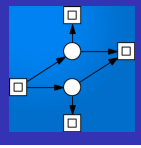
Example: 2D Biochemical Diffusion



Example: 2D Biochemical Diffusion

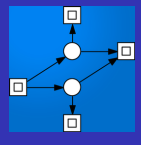


Example: 2D Biochemical Diffusion (Cont.)



- Constants definition:
  - **const** D1 = **int** with 15;
  - **const** D2 = D1;
  - **const** MIDDLE = **int** D1/2+1;
- Required colour sets:
  - **colorset** CD1 = **int** with 1 – D1;
  - **colorset** CD2 = **int** with 1 – D2;
  - **colorset** Grid2D = **Product** with CD1 x CD2;
- Variables declaration:
  - **var** x,a : CD1;
  - **var** y,b : CD2;

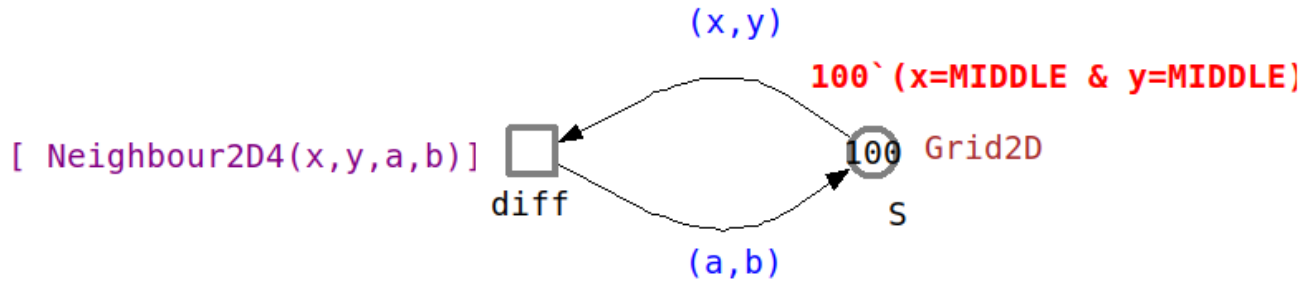
Example: 2D Biochemical Diffusion (Cont.)



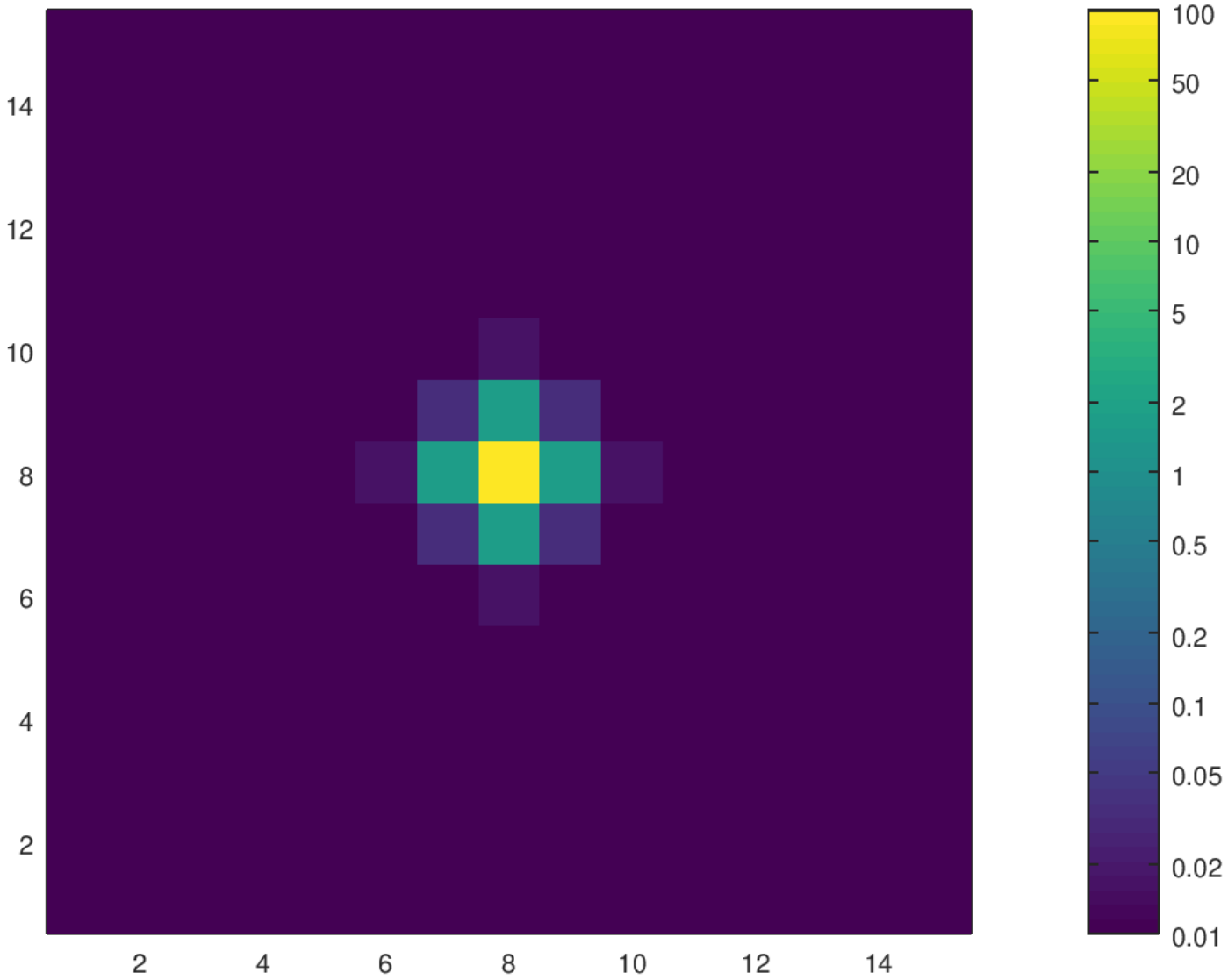
- Neighbourhood function (four-neighbour relation)

```
fun bool Neighbour2D4 (CD1 x , CD2 y , CD1 xn , CD2 yn ) {
// ( xn , yn ) is one of the up to four neighbours of ( x , y )
( xn=x & yn=y- 1) | ( xn=x & yn=y+1)
| ( yn=y & xn=x- 1) | ( yn=y & xn=x+1)
& (1<=xn & xn<=D1) & (1<=yn & yn<=D2) } ;
```

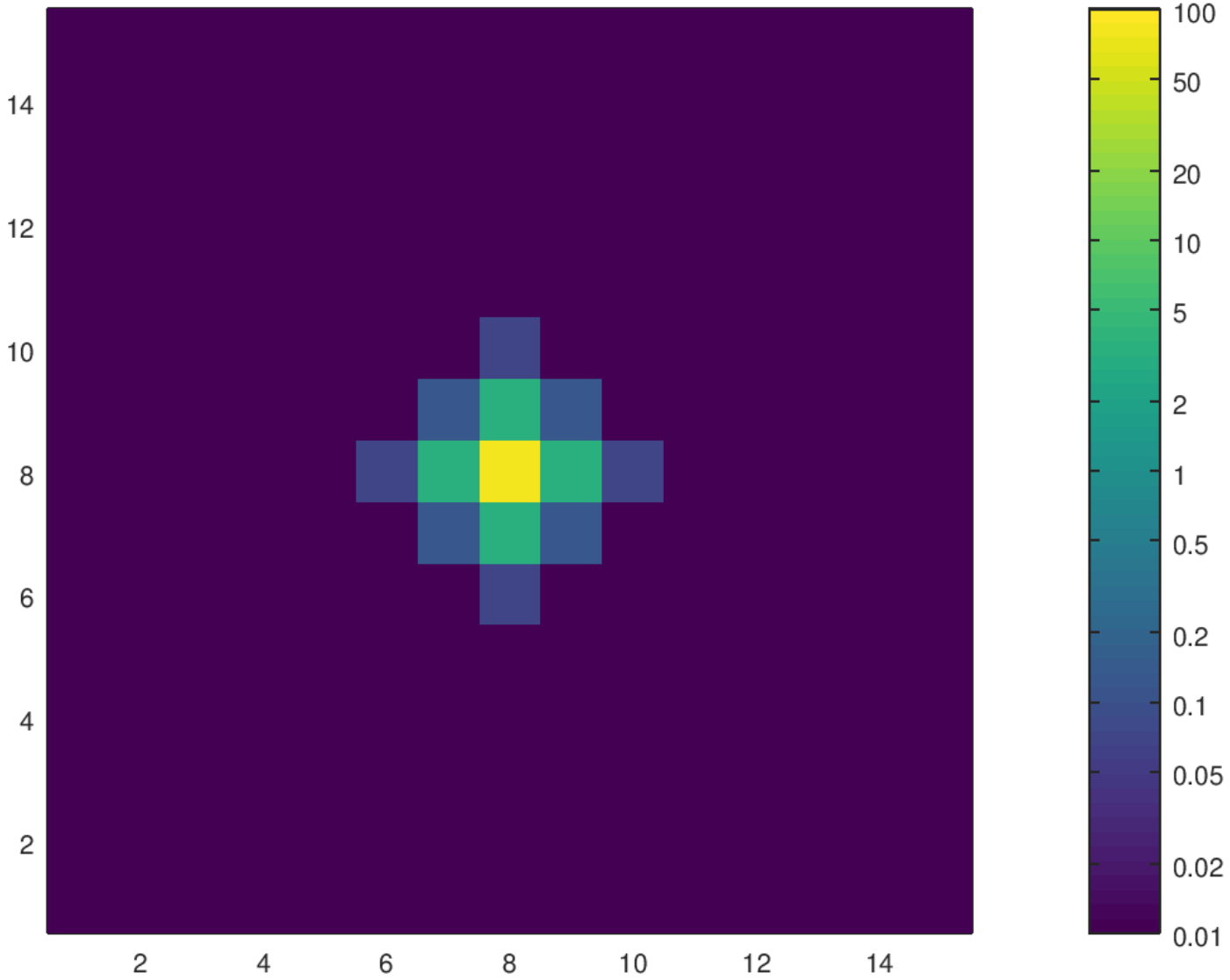
- Model



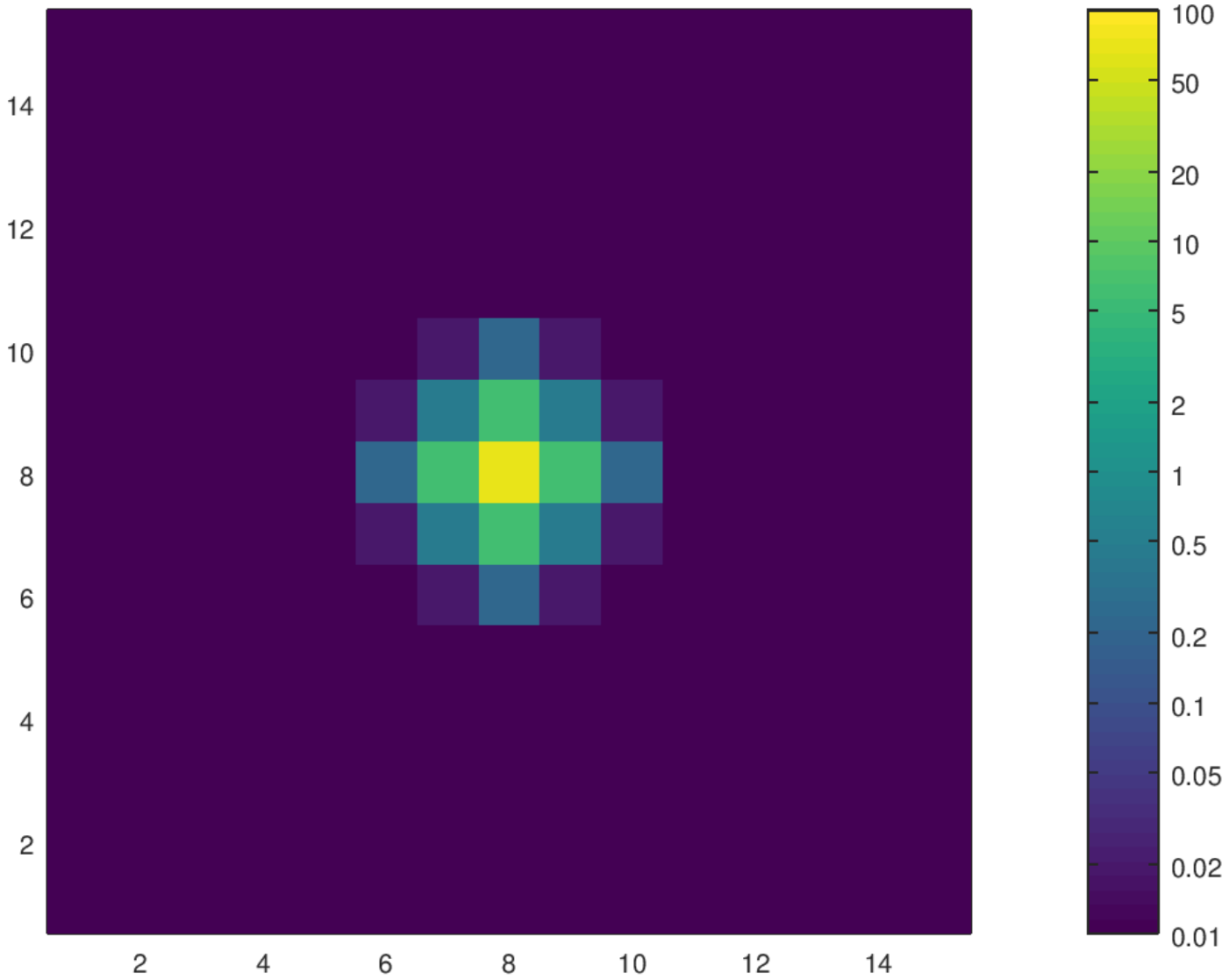
Example: 2D Biochemical Diffusion - Simulation t=20



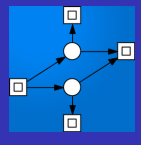
Example: 2D Biochemical Diffusion - Simulation t=40



Example: 2D Biochemical Diffusion - Simulation t=80



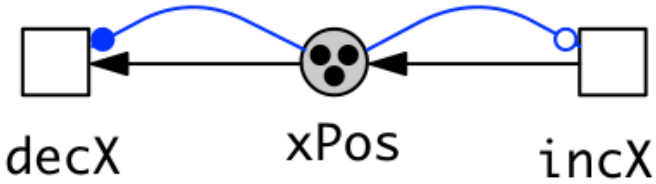
Disadvantages

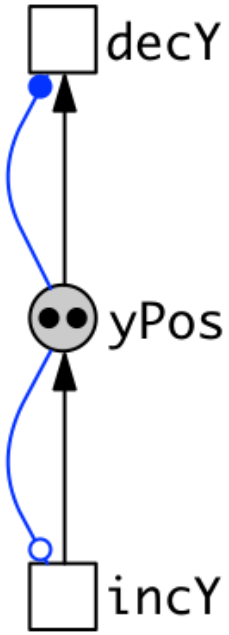


- In this encoding principle, space is modeled by discretisation into a grid and having one subnet per grid position.

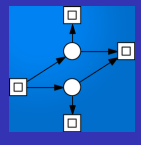
- If the entities that are moving around have internal structure (state) which has to move around as well, this leads to some issues on modelling and simulation (space exploation).

Second spatial encoding scheme

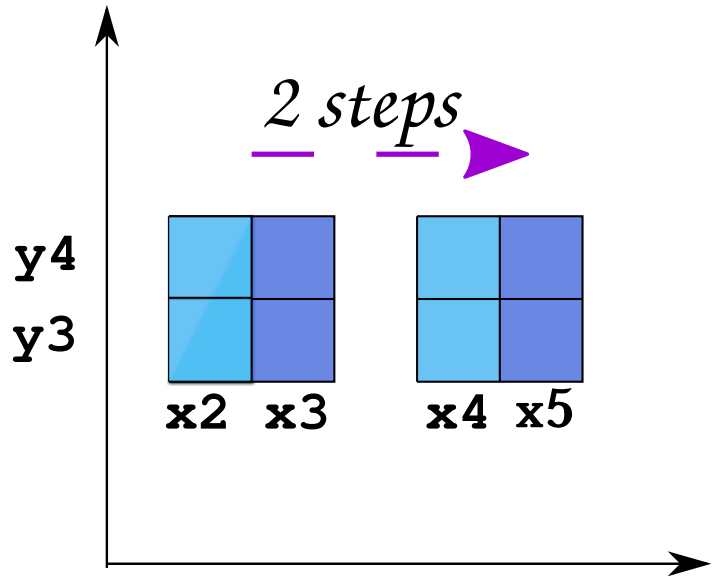




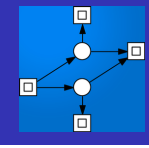
Example: Moving Robot



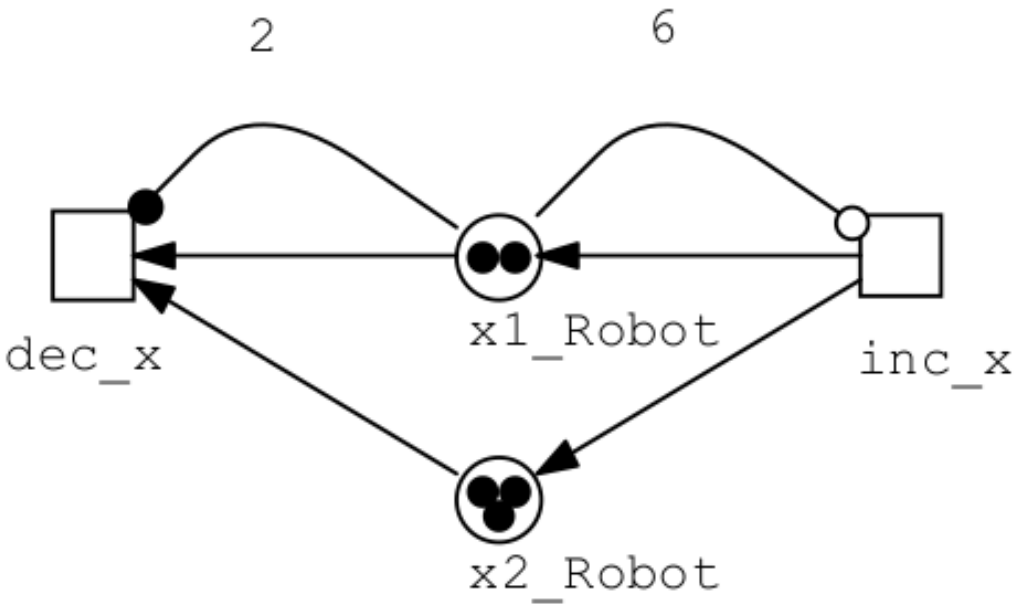
- The Robot has a size of up to 2x2 (units). It moves on a straight line in the center of the room covering a distance of 7 meters.
- Since Robot arrives the first end, it moves back in the opposite direction and so on.
- Robot initially occupies the positions x=2 and x=3 on the X-Axis and y=3 and y=4 on Y-Axis.



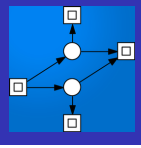
Example: Moving Robot- space encoding



- No colours.

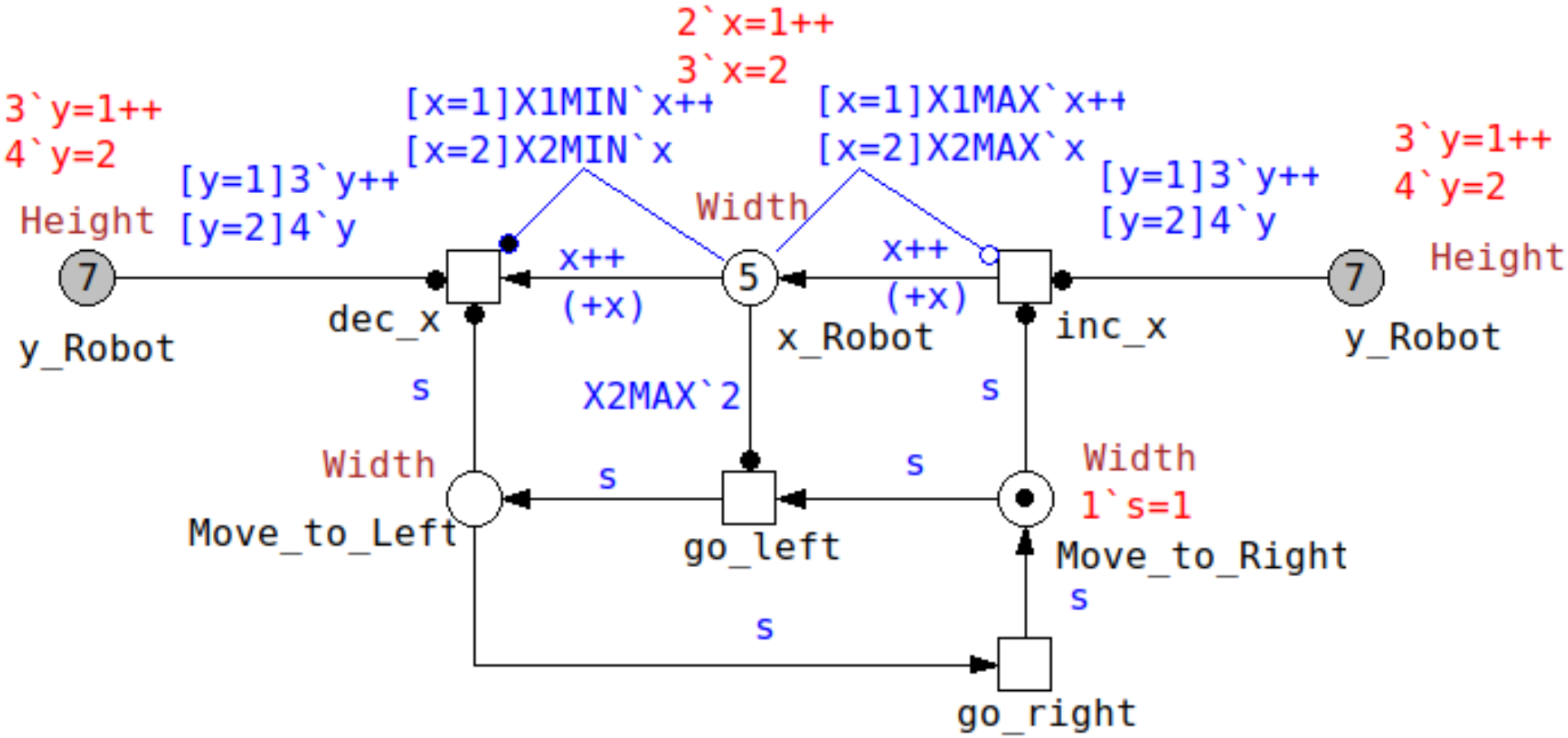


Example: Moving Robot - coloured Petri net Definitions

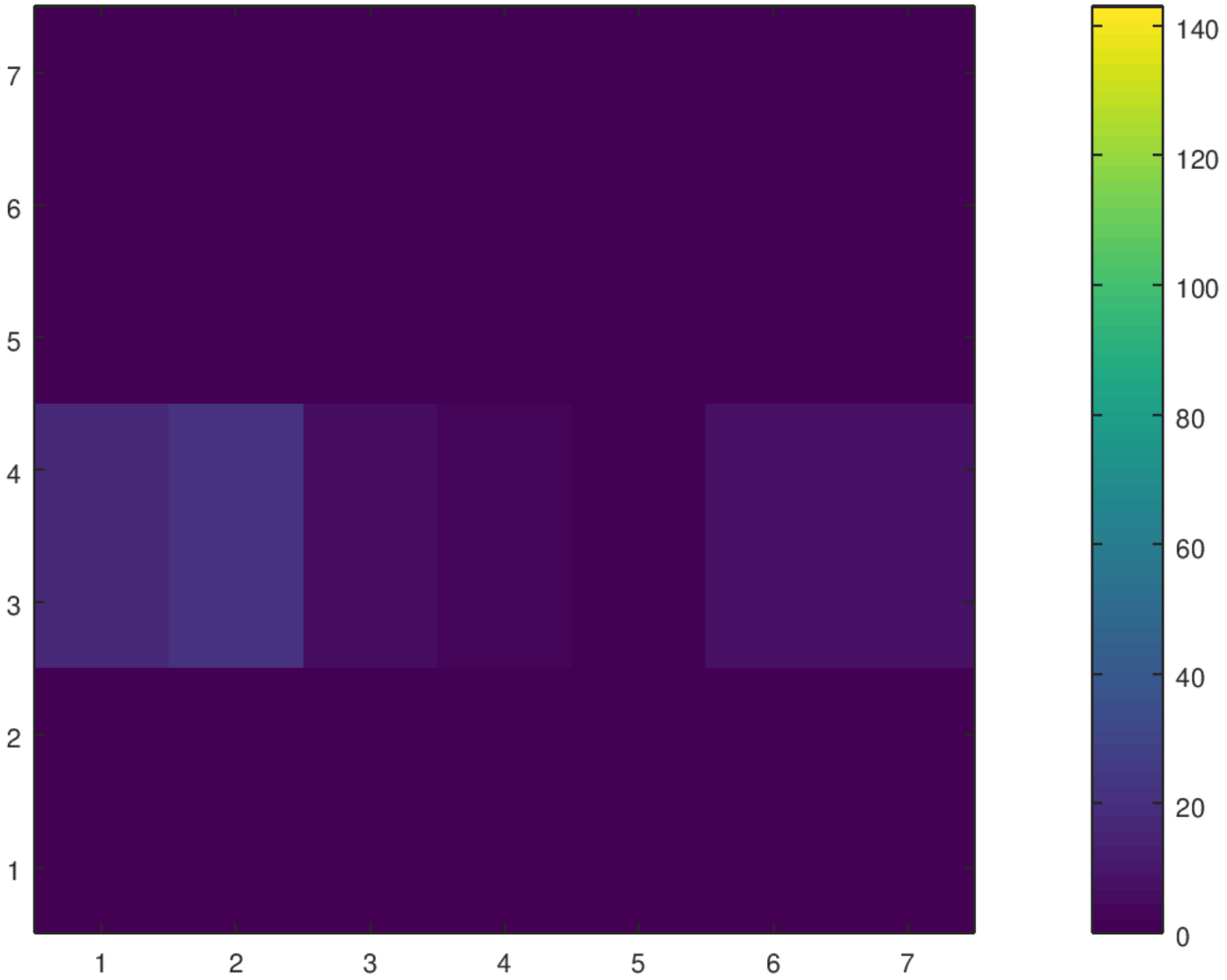


- Constants
  - **const** SIZE = int with 2;//size of Robot on each dimension
  - **const** X1MAX = int with 6;
  - **const** X2MAX = int with 7;
  - **const** X1MIN = int with 2;
  - **const** X2MIN = int with 3;
  - **const** Y1 = int with 3;
  - **const** Y2 = int with 4;
- Required colour sets
  - **colorset** Width = int with 1−SIZE;// number of occupied positions on x-axis
  - **colorset** Height = int with 1−SIZE;// number of occupied positions on y-axis
- Required Variables
  - **var** x : Width;//row index
  - **var** y : Height;//column index
  - **var** s : Width;// switching movement from right to left and vice versa.

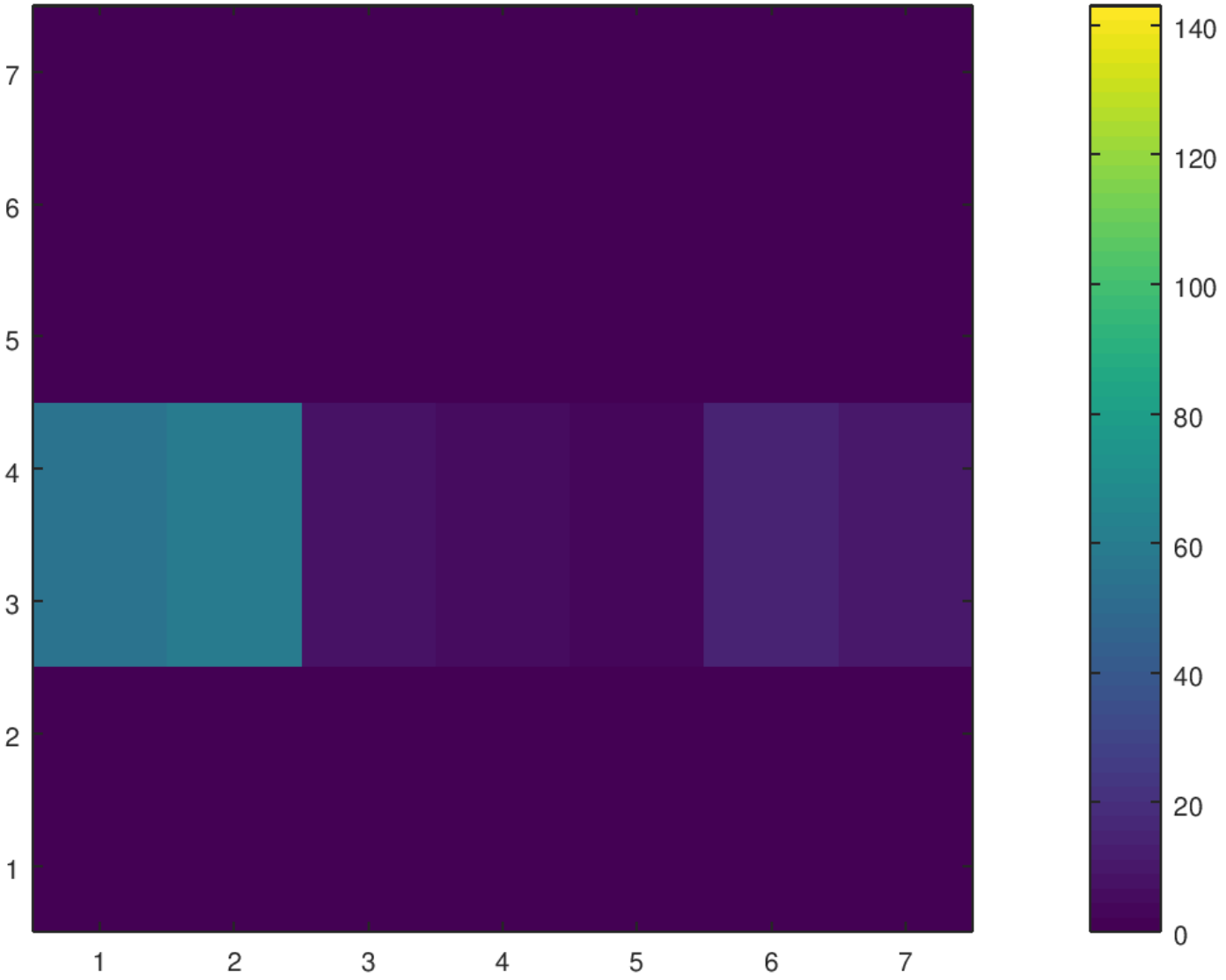
Example: Moving Robot - Model



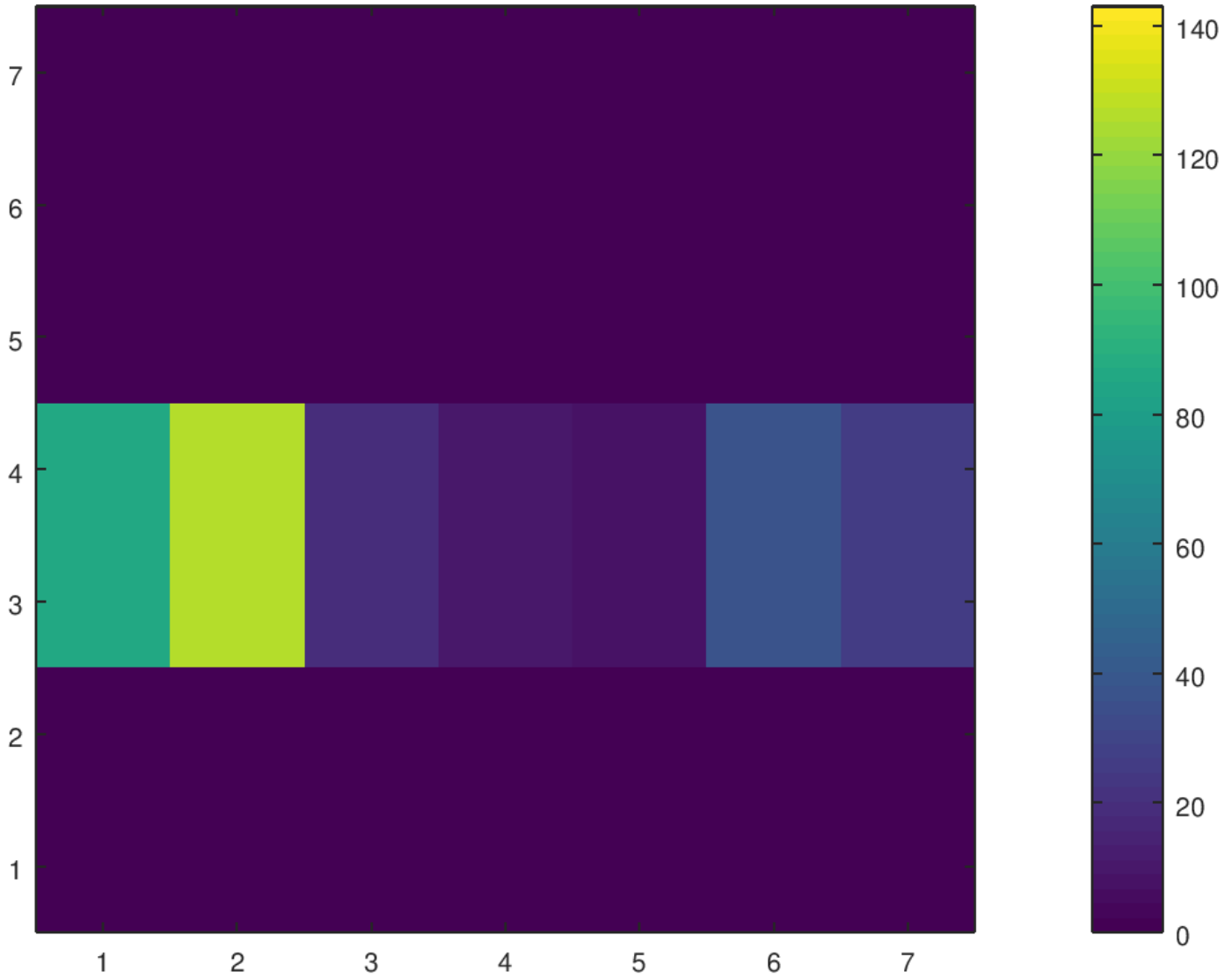
Example: Moving Robot - Model Simulation t=20s



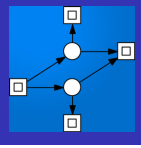
Example: Moving Robot - Model Simulation t=40s



Example: Moving Robot - Model Simulation t=80s



Spatial encoding schemes - Comparison



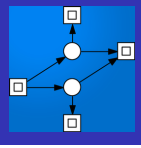
Scheme 1:

- Discretising space using finite colour sets => finite universe.

Scheme 2:

- Space is represented using coordinate places.

Spatial encoding schemes - Comparison



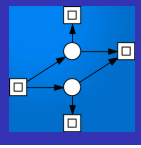
Scheme 1:

- Discretising space using finite colour sets => finite universe.
- The unfolded model increases extremely as the size of space increases.

Scheme 2:

- Space is represented using coordinate places.
- Size of the unfolded model does not dependent on size of the finite universe.

Spatial encoding schemes - Comparison



Scheme 1:

- Discretising space using finite colour sets => finite universe.
- The unfolded model increases extremely as the size of space increases.
- Local states of moving objects is not possible.

Scheme 2:

- Space is represented using coordinate places.
- Size of the unfolded model does not dependent on size of the finite universe.
- Local state of moving objects is possible.

Thank you