Transferable Knowledge in P Colonies, cP systems and QUBO

19th Brainstorming Week on Membrane Computing

Lucie Ciencialová[1], Michael Dinneen[2],
Luděk Cienciala[1], Radu Nicolescu[2]

January 27, 2023

1 Institute of Computer Science, Silesian University in Opava, Czech Republic
Institute of Computer Science and Research Institute of the IT4Innovations Centre of Excellence, Silesian University in Opava, Czech Republic
lucie.ciencialova@fpf.slu.cz

2 Department of Computer Science, University of Auckland

Outline

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

P Colonies, 2D P Colonies

- Concept
- Rules
- Computation

Transferable knowledge

- Idea
- P Colony Example
- 2D P Colony Example

Conclusion

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

P Colonies

Introduction

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

P colonies

Introduced by Gheorghe Păun, Alica Kelemenová and Jozef Kelemen in 2000[1]. The system consists of **agents** - one membrane cells, each containing **multiset of objects** of given cardinality and they are equipped with a **set of programs**.

[1] J. Kelemen, A. Kelemenová, and Gh. Păun. "Preview of P colonies: A biochemically inspired computing model". In: *Workshop and Tutorial Proceedings. Ninth International Conference on the Simulation and Synthesis of Living Systems (Alife IX)*. Boston, Mass, 2004, pp. 82–86.

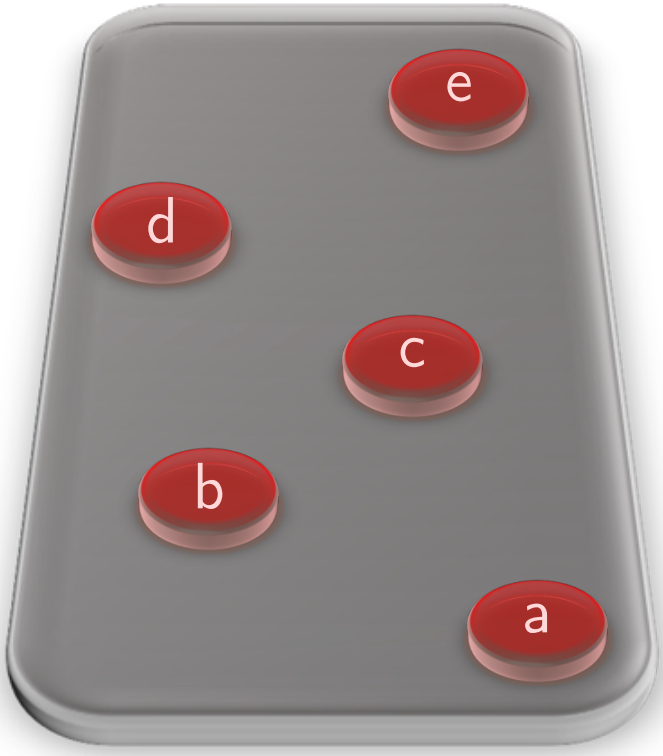
Introduction

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

P colonies

Agent

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA



- collection of objects
- embedded in a membrane
- with set of programs

Introduction

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Rules

- Rewriting rule $a \longrightarrow b$

  Every agent can change its content by evolving it.

  Agent:abc Rule:$a \longrightarrow b$ Agent:bbc

  Env.: bbd Env.: bbd

- communication rule $c \longleftrightarrow d$

  Every agent can change its state and content of environment by communication.

  Agent:abc Rule:$c \longleftrightarrow d$ Agent:abd

  Env.: bbd Env.: bbc

Capacity of cell

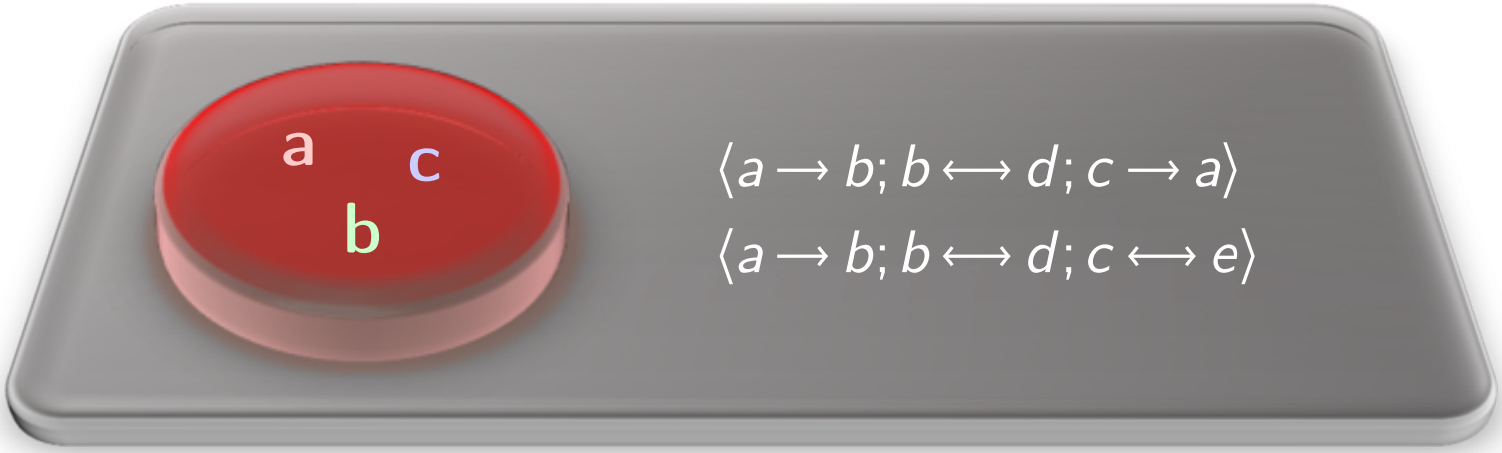
SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

- The number of objects inside the cell is constant during a computation.
- All cells have the same number of objects inside them
- This number is called **capacity** of P Colony
- A capacity sets the number of rules in program

Capacity of cell

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

- The number of objects inside the cell is constant during a computation.
- All cells have the same number of objects inside them
- This number is called **capacity** of P Colony
- A capacity sets the number of rules in program

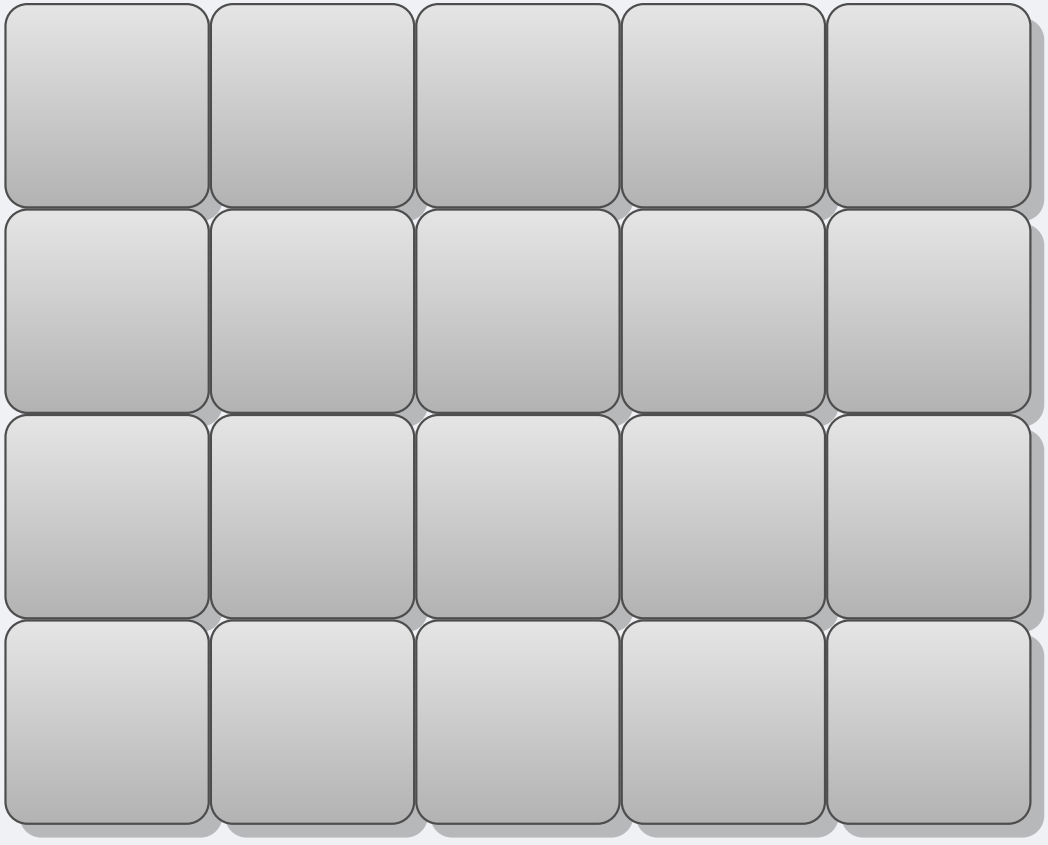


2D P Colonies

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Environment in a form of 2D grid of square cells.

Environment and agents

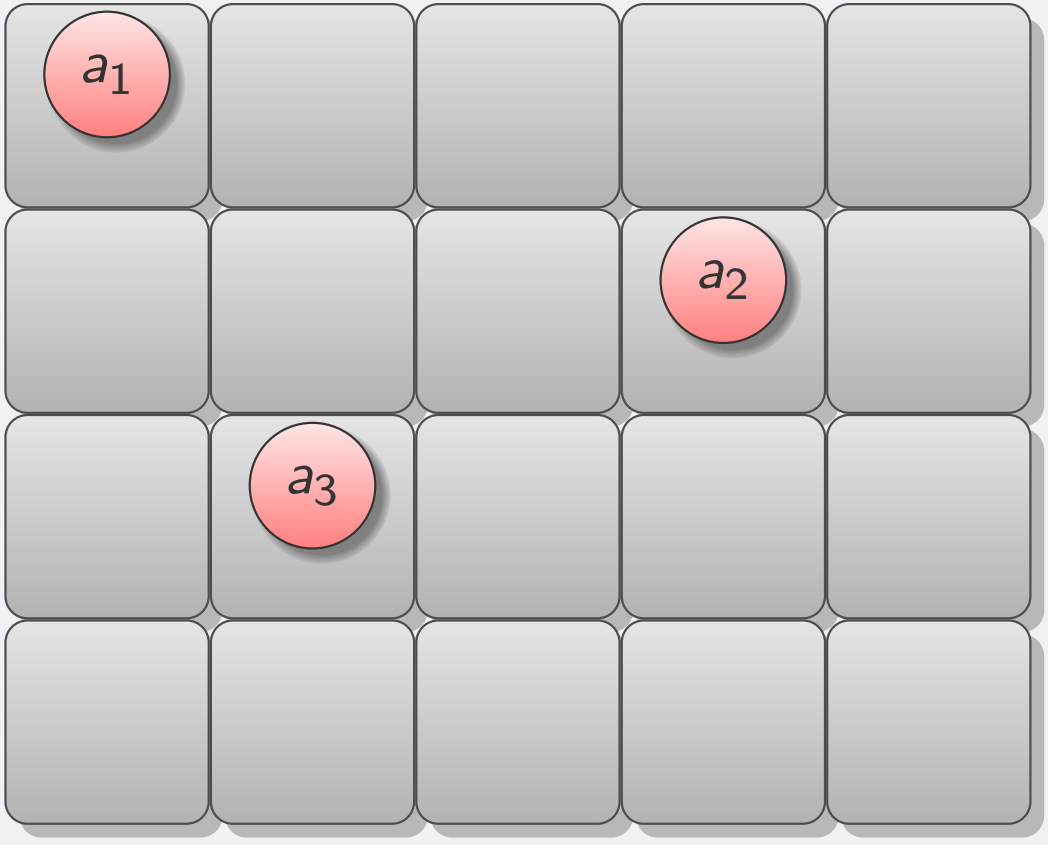


2D P Colonies

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Environment in a form of 2D grid of square cells.

Environment and agents

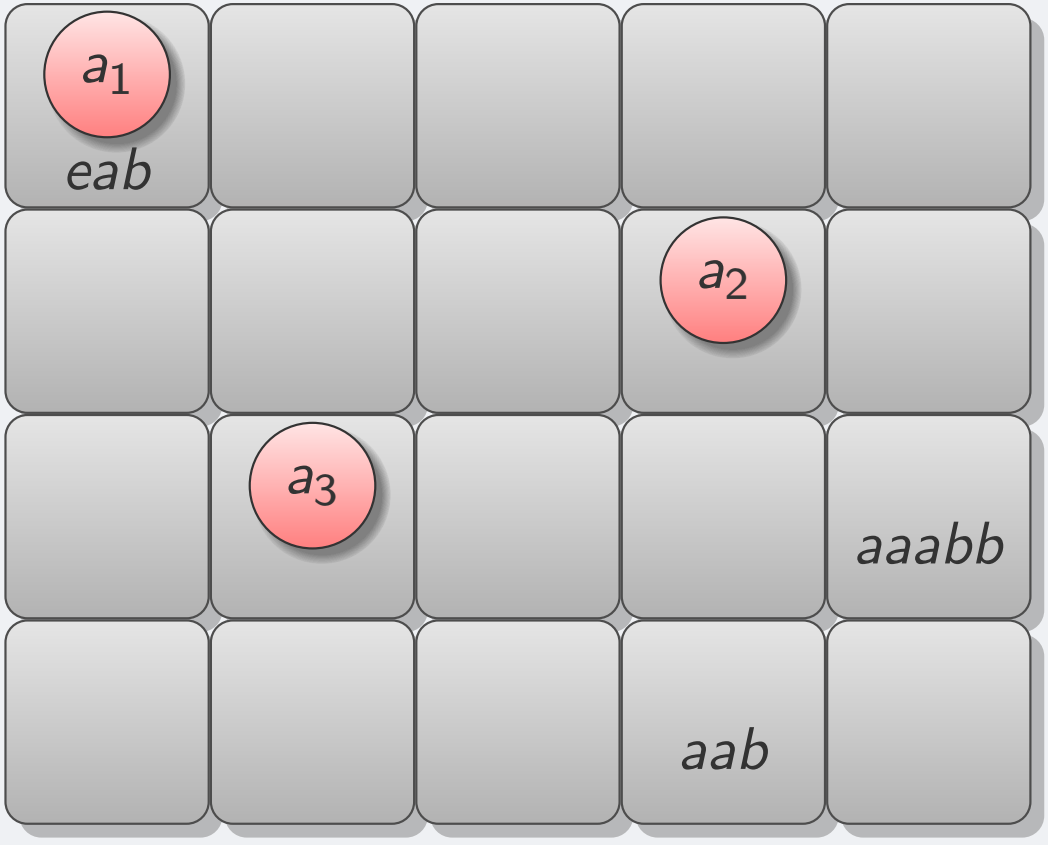


2D P Colonies

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Environment in a form of 2D grid of square cells.

Environment and agents

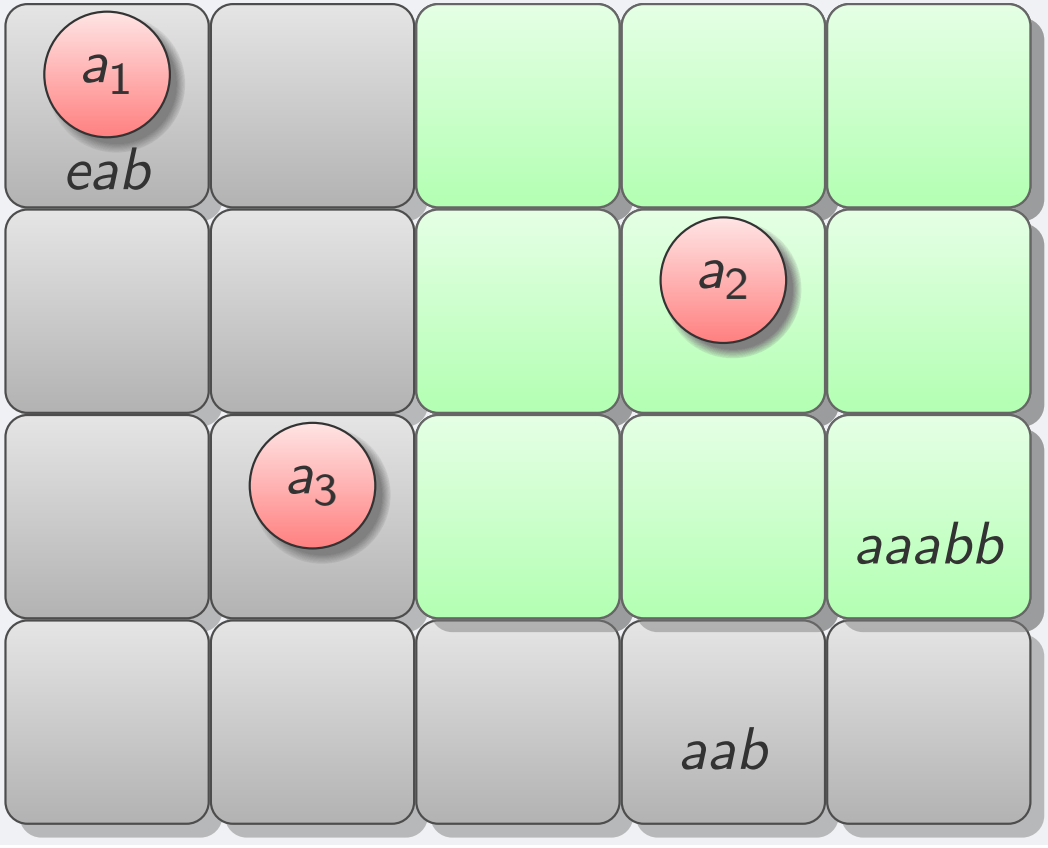


2D P Colonies

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Environment in a form of 2D grid of square cells.

Environment and agents

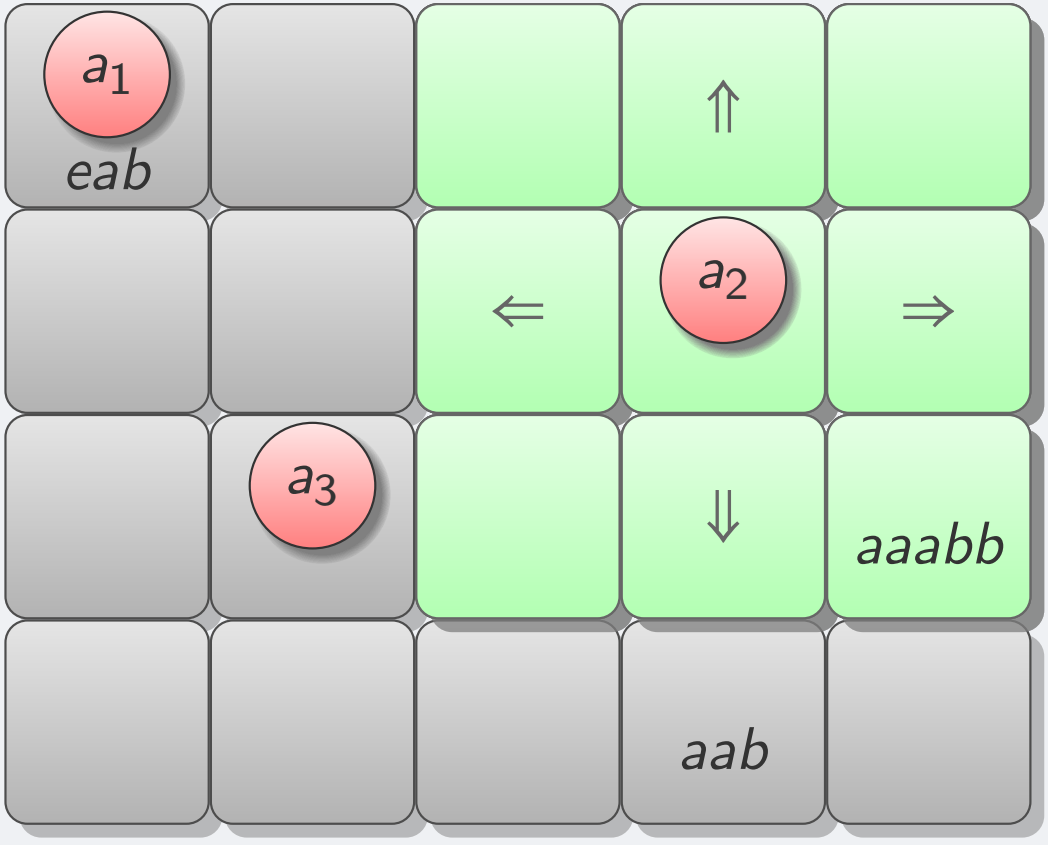


2D P Colonies

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Environment in a form of 2D grid of square cells.

Environment and agents



SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

Transferable Knowledge

Transferable programs

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

A transferable program is an ordered pair

$$(\text{program}, \text{condition})$$

We distinguish two kinds of conditions:

1. **Object condition** - for a program to be transferred, the destination must contain (or must not contain) certain objects. A condition is specified as a set of multisets of objects. Each of the multisets has a size equal to the capacity P of the colony.
2. **Program condition** - for a program to be transferred, the destination must contain (or not contain) a certain program.

Transferable programs - example

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

Let Π be a P colony of capacity 2 with one agent and working with objects a, b, e, f.

Example

$(\langle a \to b;\ a \longleftrightarrow e \rangle; \{aa\})$

– when such program is placed in the environment an agent can consume it only if it contains two copies of object a;

– when such a program is placed in the agent it can be transferred into environment only if the environment contains at least two copies of object a;

Transferable programs - example

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Let Π be a P colony of capacity 2 with one agent and working with objects a, b, e, f.

Example

$(\langle a \rightarrow b;\ a \longleftrightarrow e \rangle; \{\neg ee, \neg be\})$

– when such program is placed in the environment an agent can consume it only if it does not contain two copies of object e or one object b and one object e;

– such a program cannot be transferred into the environment. The conditions can never be met, the environment always contains environmental objects.

Transferable programs - example

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

Let Π be a P colony of capacity 2 with one agent and working with objects a, b, e, f.

Example

$(\langle a \to b;\ a \longleftrightarrow e\rangle; \{\langle e \to a;\ e \longleftrightarrow a\rangle\})$

– this program can only be moved if program $\langle e \to a;\ e \longleftrightarrow a\rangle$ is contained inside the recipient (in its program set if it is an agent, or directly in the environment if the program is to be moved there)

Transferable programs

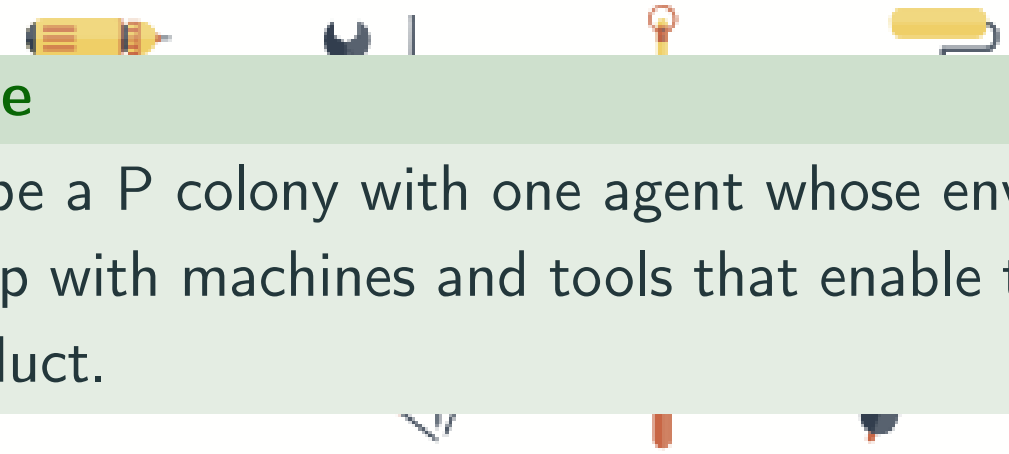
SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

We call a transferable program (program; condition) *permanent* (denoted by the p in the subscript) if each time the program is moved to a different destination, one copy of the program remains at the original location.

Such programs can be considered *knowledge sources* (and in some cases object sources).

P Colony with transferable programs

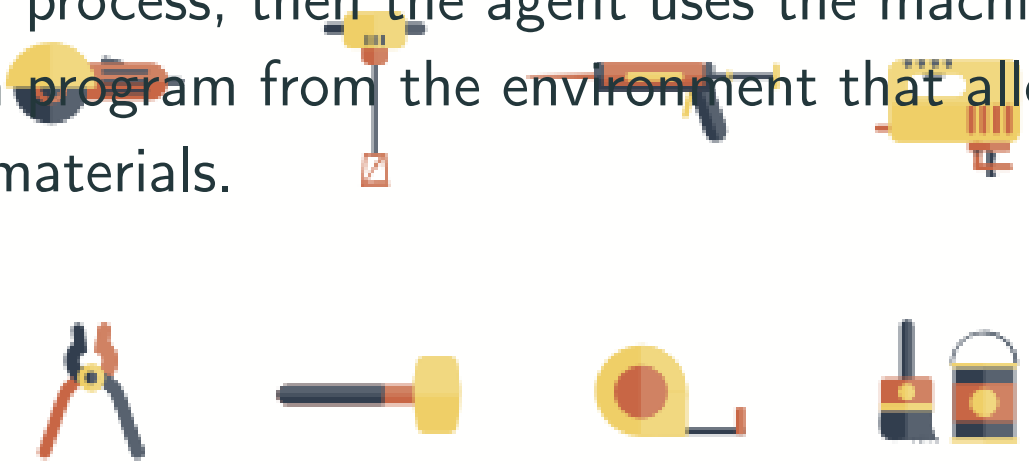
SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA



Example

Let Π_1 be a P colony with one agent whose environment is a workshop with machines and tools that enable the processing of the product.

If the agent has (contains) raw materials that the machine (or tool) can process, then the agent uses the machine. That is, it obtains a program from the environment that allows it to process the raw materials.



P Colony with transferable programs

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

P colony $\Pi_1 = \left(A, e, \boxed{tp}, v_e, B\right)$

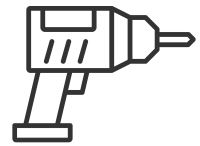
Alphabet

pow	piece of wood,	*tth*	table top with holes,
p	paint,	*ttn*	table top with nuts,
4n	four nuts,	*tt1l*	table top with 1 leg,
s-b	double ended screw bolt,	*tt2l*	table top with 2 legs,
l	leg,	*tt3l*	table top with 3 legs,
lh	leg with hole,	*tc*	complete table,
lc	complete leg,	*ts*	sanded table,
tt	table top,	*tp*	painted table,

P Colony with transferable programs

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

P colony $\Pi_1 = \left(A, e, \boxed{tp}, v_e, B\right)$



Initial contents and transferable programs

$v_e =$ $\boxed{pow}$ $\boxed{pow}$ $\boxed{pow}$ $\boxed{pow}$ $\boxed{pow}$ $\boxed{4n}$ $\boxed{p}$ $\boxed{s\text{-}b}$ $\boxed{s\text{-}b}$ $\boxed{s\text{-}b}$ $\boxed{s\text{-}b}$

$\left(\left\langle \boxed{pow} \rightarrow \boxed{tt}, e \longleftrightarrow e \right\rangle, \{\boxed{pow}\ e\}\right)$ saw

$\left(\left\langle \boxed{pow} \rightarrow \boxed{l}, e \longleftrightarrow e \right\rangle, \{\boxed{pow}\ e\}\right)$ wood lathe

$\left(\left\langle \boxed{tt} \rightarrow \boxed{tth}, e \longleftrightarrow e \right\rangle, \{\boxed{tt}\ e\}\right)$ drill

$\left(\left\langle \boxed{l} \rightarrow \boxed{lh}, e \longleftrightarrow e \right\rangle, \{\boxed{l}\ e\}\right)$ drill

$\left(\left\langle \boxed{tc} \rightarrow \boxed{ts}, e \longleftrightarrow e \right\rangle, \{\boxed{tc}\ e\}\right)$ angle grinder

$\left(\left\langle \boxed{ts} \rightarrow \boxed{tp}, p \rightarrow e \right\rangle, \{\boxed{ts}\ p\}\right)$ paint spray gun

P Colony with transferable programs

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Agent B is in the initial state ee and its set of programs is formed from its ability to work with or without hand tools:

Agents programs

$$
\begin{aligned}
P = \{ & \left\langle e \longleftrightarrow \boxed{pow}, e \longleftrightarrow e \right\rangle, && \text{take a piece of wood} \\
& \left\langle \boxed{lh} \rightarrow \boxed{lh}, e \longleftrightarrow \boxed{s\text{-}b} \right\rangle, && \text{take a screw bolt} \\
& \left\langle \boxed{lh} \rightarrow \boxed{lc}, \boxed{s\text{-}b} \rightarrow e \right\rangle, && \text{mount the screw bolt} \\
& \left\langle \boxed{lc} \longleftrightarrow, e \longleftrightarrow e \right\rangle, && \text{put down complete leg} \\
& \left\langle \boxed{tth} \rightarrow \boxed{tth}, e \longleftrightarrow \boxed{4n} \right\rangle, && \text{take four nuts} \\
& \left\langle \boxed{tth} \longleftrightarrow \boxed{ttn}, \boxed{4n} \rightarrow e \right\rangle, && \text{put nuts on the holes} \\
\vdots & \\
& \left\langle \boxed{ttn} \longleftrightarrow \boxed{tt1l}, \boxed{lc} \rightarrow e \right\rangle, && \text{mount the complete leg on the table top}
\end{aligned}
$$

2D P Colony with transferable programs

2D P Colony with transferable programs

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

Example

Consider a 2D P colony with two special agents helpers and a ot of regular ones. The first agent can move around the environment and spread the transferable program allowing the agent to consume slave object. The regular agent's programs allows the agent to move if slave object is inside the agent. If the regular agent enters position in the environment where the second helper is, it can import a program that enables the agent to make its own object s. The agent that imports such a program will then not have to follow the first agent and can move freely.

2D P Colony with transferable programs

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

2D P colony $\Pi_1 = \left(A, e, \boxed{tp}, v_e, B \right)$

Example

Imagine a student moving within the school building to acquire the necessary knowledge and pass an exam. We can simulate such a situation by using a 2D P colony.

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

cP systems and QUBO

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

cP system = P system with compound terms[2]

[2]Radu Nicolescu and Alec Henderson. "An Introduction to cP Systems". In: *Enjoying Natural Computing: Essays Dedicated to Mario de Jesús Pérez-Jiménez on the Occasion of His 70th Birthday*. Ed. by Carmen Graciani et al. Cham: Springer International Publishing, 2018, pp. 204–227. isbn: 978-3-030-00265-7. doi: 10.1007/978-3-030-00265-7_17.

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

Formally, a cP system is a construct

$$\Pi = (T, A, O, C, R, S, \overline{s}), \text{ where}$$

T is the set of top-level cells at the start of the evolution of the system; A is the alphabet of the system; O is the set of multisets of initial objects in the top-level cells; C is the set of sets of channel endpoint labels for inter-top-level cell communication that can be found in each top-level cell; R is the set of rule-sets for each top-level cell; S is the set of possible states of the top-level cells; and $\overline{s} \in S$ is the starting state of every top-level cell in the system.

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

QUBO problem

Quadratic Unconstrained Binary Optimisation is an NP-hard mathematical optimization problem.

cP systems and QUBO

QUBO problem

Integer version of the problem of minimizing a quadratic objective function

$$x^* = \min_{\vec{x}} \vec{x}^T Q \vec{x}$$

where:

- $\vec{x}$ is a n-vector of binary (Boolean) variables $x_i \in \{0;1\}, 0 \leq i \leq n-1$
- $n \in \mathbb{N}_0$ - the number of variables in $\vec{x}$
- $i, j \in \mathbb{N}_0$
- Q is an upper-triangular $n \times n$ matrix where $q_{i,j} \in \mathbb{Z}, 0 \leq i \leq j \leq n-1$ are possibly non-zero coefficients

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

QUBO problem

Formally, QUBO problems are of the form:

$$x^* = \min_{\vec{x}} \sum_{i \leq j} x_i q_{i,j} x_j, \quad \text{where } x_i, x_j \in \{0, 1\}$$

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

For every integer a there is

$$i(x,y),\ \text{where } x,y \in \mathbb{N}_0 \iff a = x - y$$

For example:

$i(1,4) \iff -3 = 1-4$
$i(5,8) \iff -3 = 5-8$
$i(8,11) \iff -3 = 8-11$
$i(0,3) \iff -3 = 0-3$

Every representation $i(x,y)$ can be converted into canonical form:

$$i(x,y) \sim \begin{cases} i(x',0) & \text{for } x \geq y \quad \text{where } x' = x-y \\ i(0,y') & \text{for } x < y \quad \text{where } y' = y-x \end{cases}$$

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

We developed a cP system that finds minimal value of a QUBO in three phases of computation.

1. In the first phase, all possible values assignment is generated.
2. The second phase is devoted to generating of all polynomials.
3. In the third phase, related coefficients are added together to evaluate potential solutions for the assignments produced from phases 1 and 2.

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

Input:

- For every variable x_i storing value $y_i \in \{0,1\}$ there is complex object
$$a\big(in(i)val(y_i')\big) \text{ where } y_i' \in \{\lambda, 1\}.$$
- For every coefficient $q_{i,j}$ there is complex object
$$q\,(in1(i)in2(j)val(+(x)\ -(y))) \text{ where } q_{i,j} = x - y.$$
- Two counters (counter-like objects): $C_1(\lambda), C_2(n)$.
- Empty list of values of variables: $I(C_1(\lambda))$ with counter $C_1(\lambda)$ inside.

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

$$S_1 \quad C_2(1X) \longrightarrow_{\min} S_2 \quad v(\lambda)v(1)C_2(X) \qquad (1)$$

Skin membrane contains complex object $C_2(n) - n = 1^n$

Unified rule:

$$S_1 \quad C_2(11^{n-1}) \longrightarrow_{\min} S_2 \quad v(\lambda)v(1)C_2(1^{n-1})$$

By the execution of the rule (1), two complex objects - $v(\lambda)$ and $v(1)$ are generated and the number of 1s inside $C_2()$ is decreased by one.

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

$$S_2 \quad I(\_) \longrightarrow_{\max} S_1 \tag{3}$$

$$S_2 \quad v(\_) \longrightarrow_{\max} S_1 \tag{4}$$

The rules (3) and (4) can erase all objects $I()$ and $v()$ inside the cell.

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

$$S_2 \quad \longrightarrow_{\max} S_1 \quad I(a(in(X)val(Y))\, C_1(X1)Z) \mid I(C_1(X)Z) \mid v(Y) \qquad (2)$$

For every combination of X, Z and Y there can be one unified rule.

- X is the same for all objects in cP system at current step of computation,
- Z is unique for every object $I()$,
- there are two possible evaluations for Y

There is $1 \times |I()| \times 2$ rules that can be executed in parallel.

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

$$S_2 \quad \longrightarrow_{\max} S_1 \quad I(a(in(X)val(Y))\, C_1(X1)Z) \qquad \begin{array}{l} \mid I(C_1(X)Z) \\ \mid v(Y) \end{array}$$

S_2 : $I(C_1(\lambda))C_2(n-1)$ $v(\lambda)v(1)$ example configuration

$(2)_1$ $X=\lambda, Y=\lambda, Z=\lambda$

$$S_2 \quad \longrightarrow_{\max} \quad S_1 \quad I(a(in(\lambda)val(\lambda))\, C_1(1)) \quad \begin{array}{l} \mid I(C_1(\lambda)) \\ \mid v(\lambda) \end{array}$$

$(2)_2$ $X=\lambda, Y=1, Z=\lambda$

$$S_2 \quad \longrightarrow_{\max} \quad S_1 \quad I(a(in(\lambda)val(1))\, C_1(1)) \quad \begin{array}{l} \mid I(C_1(\lambda)) \\ \mid v(1) \end{array}$$

cP systems and QUBO

SILESIAN UNIVERSITY
FACULTY OF PHILOSOPHY AND SCIENCE IN OPAVA

(3)

$S_2 \quad I(C_1(\lambda)) \longrightarrow_{\max} \quad S_1$

$(4)_1$

$S_2 \quad v(\lambda) \longrightarrow_{\max} \quad S_1$

$(4)_2$

$S_2 \quad v(1) \longrightarrow_{\max} \quad S_1$

$S_1 : \quad I(a(in(\lambda)val(\lambda))\, C_1(1))\ \ I(a(in(\lambda)val(1))\, C_1(1))\, C_2(n-1)$

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

The second phase

The idea of the second phase is to generate objects $p()$, which will contain representatives of quadratic elements that will be multiplied by the coefficients of one (say the i-th) row of the matrix Q. However, if the value of the variable x_i is zero, the generation of the row is omitted since its value will be zero.

cP systems and QUBO

SILESIAN
UNIVERSITY
FACULTY OF PHILOSOPHY
AND SCIENCE IN OPAVA

The third phase

In the third phase, for each combination of non-zero values of $x_i x_j$, we will add the value of the coefficient q_{ij} to the result in the object $I()$. After that we convert all values to canonical form. If there is at least one negative number we search for the maximum of negative numbers, if there is no negative number we search for the minimum of positive numbers (and zero).

Questions? Comments?

Thank you for your attention!