ON THE EFFICIENT SIMULATION OF POLARIZATIONLESS P SYSTEMS WITH ACTIVE MEMBRANES

Zsolt Gazdag
Institute of Informatics
University of Szeged

Gábor Kolonits
Faculty of Informatics,
Eötvös Loránd University

17th BWMC, February 5, 2018 - February 8, 2018, Seville, Spain

MOTIVATION

- Pauns's conjecture:

  > Active membranes without polarization and non-elementary division characterise the class **P**.

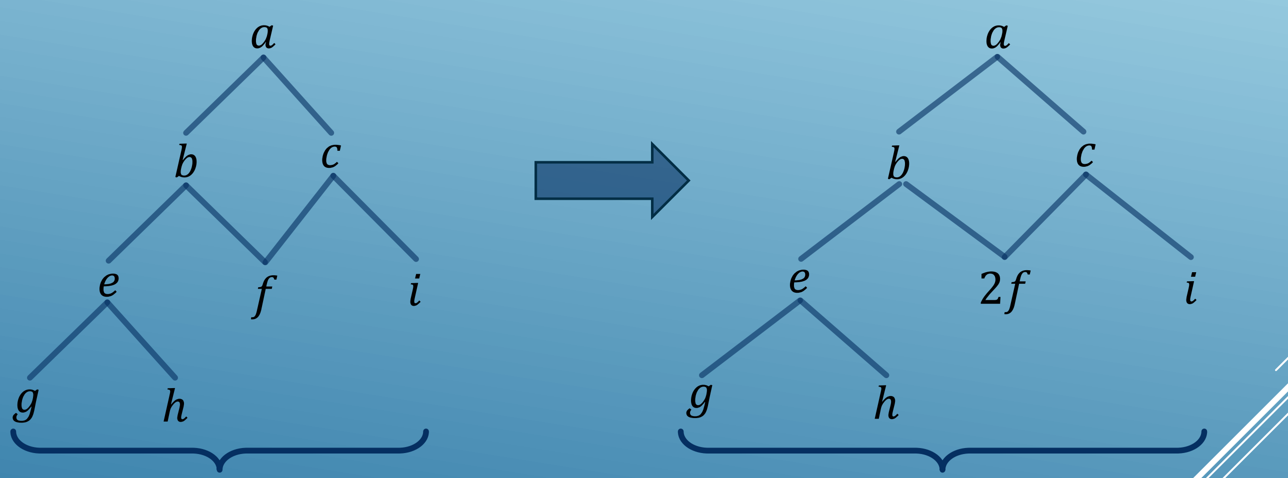
- Challenging to prove even if just division or dissolution rules are allowed
  - Proved in [2009, Woods, Murphy, Pérez-Jiménez, Riscos-Núñez]
    - Object division graphs are used
    - A particular computation is simulated
    - Only "important" membranes are explicitly simulated
- Our aim is a representation with witch we can answer the following question efficiently:
  - Which and how many objects are released to the parent membrane by the elementary membranes during a particular step of the system?

PRELIMINARIES

- We consider restrictied P systems with active membranes:
  - no polarization
  - the membrane structure is linearly nested,
  - only division and dissolution rules are employed,
  - the computations are confluent, and
  - no object can divide and dissolve the same membrane
    - E.g. $[a]_h \to [b]_h[c]_h$ and $[a]_h \to d$ is not possible
- We simulate a particular computation:
  - division has priority over dissolution
    - E.g. $[a]_h \to [b]_h[c]_h$, $[e]_h \to f$ implies $[a\ e]_h \Rightarrow [b\ e]_h[c\ e]_h \Rightarrow b\ f\ c\ f$
  - similar rules are lexicographically ordered
    - E.g. $[a]_h \to [b]_h[c]_h$ and $[e]_h \to [f]_h[g]_h$ implies $[a\ e]_h \Rightarrow [b\ e]_h[c\ e]_h \Rightarrow$ $[b\ f]_h\ [b\ g]_h\ [c\ f]_h\ [c\ g]\_h$

PRELIMINARIES

- Object division graph (odg)
  - Example:
    - Rules: $[a] \to [b][c],\ b \to [e][f],\ \ c \to [f][i],\ [e] \to [g][h]$
    - No division by the rest of the objects

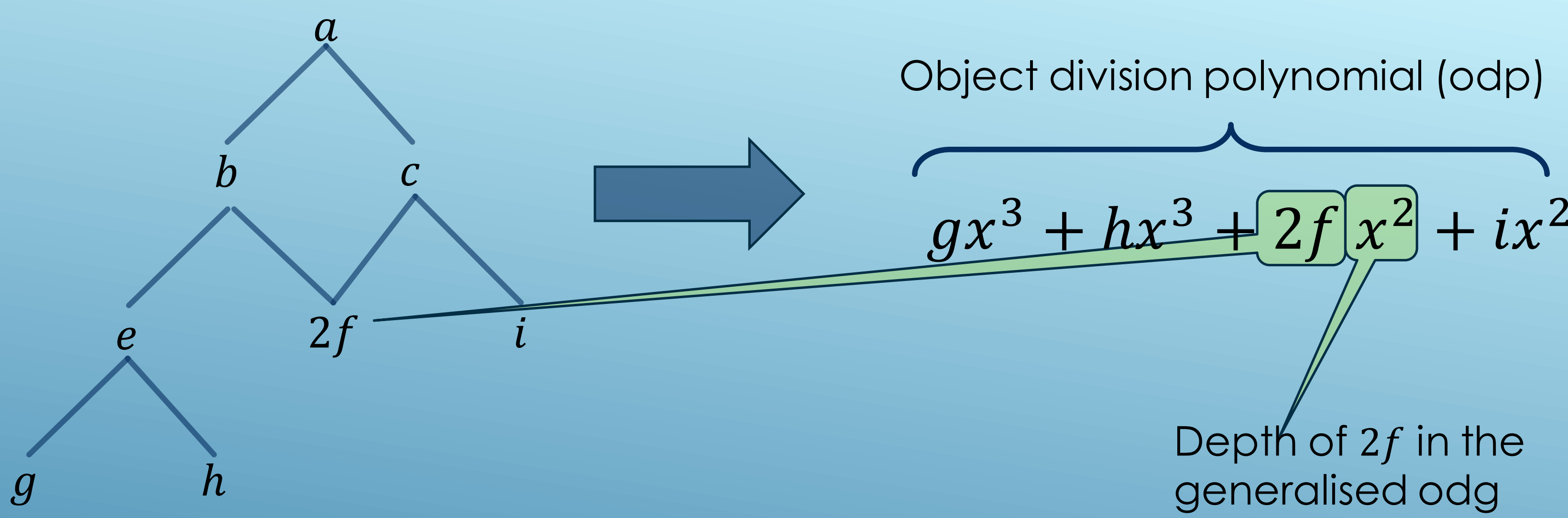


Object division graph

Generalised odg (godg)

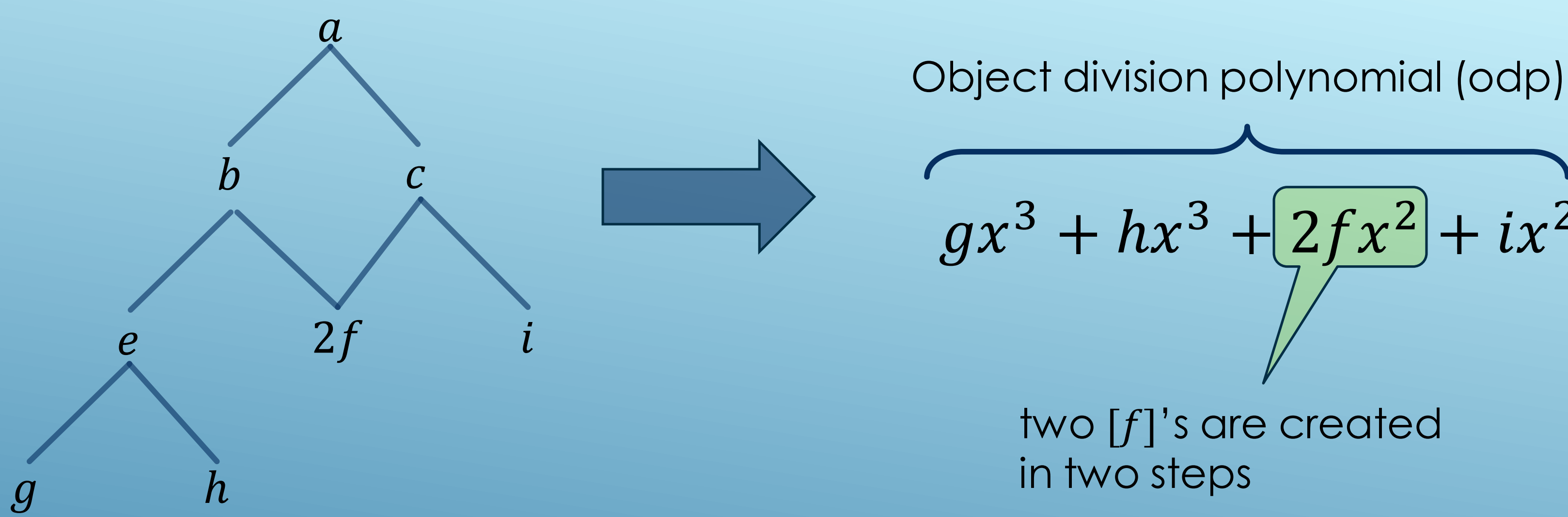
- Can generalised odg's be efficiently calculated?

THE POLYNOMIAL REPRESENTATION



- Notice: Labels of inner nodes are not needed to give the polynomial
- The divisions: $[a] \Rightarrow [b][c] \Rightarrow [e][f]\ [f][i] \Rightarrow [g][h][f]\ [f][i]$

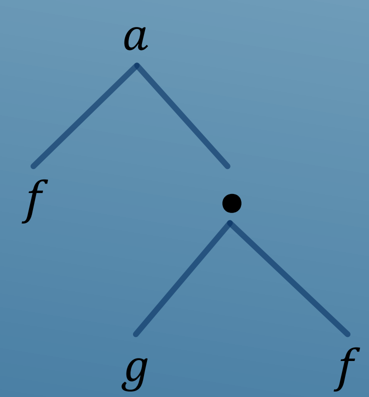
THE POLYNOMIAL REPRESENTATION



- Notice: Labels of inner nodes are not needed to give the polynomial
- The divisions: $[a] \Rightarrow [b][c] \Rightarrow [e][f]\ [f][i] \Rightarrow [g][h][f]\ [f][i]$
- The two [f]'s are created in two steps
- We can learn this also from the polynomial

THE POLYNOMIAL REPRESENTATION

godg's:

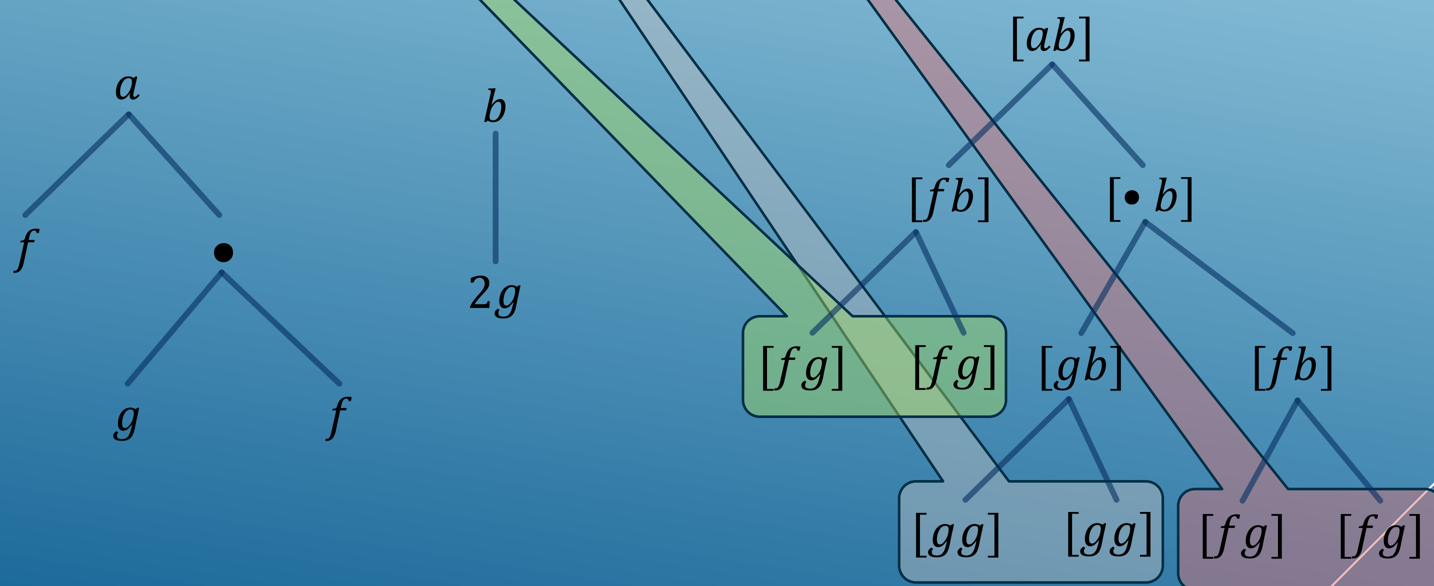


odp's: $fx + gx^2 + fx^2$ $\qquad$ $2gx$

- Let's multiply the odp's
  - $(fx + gx^2 + fx^2) \cdot 2gx = 2fgx^2 + 2ggx^3 + 2fgx^3$

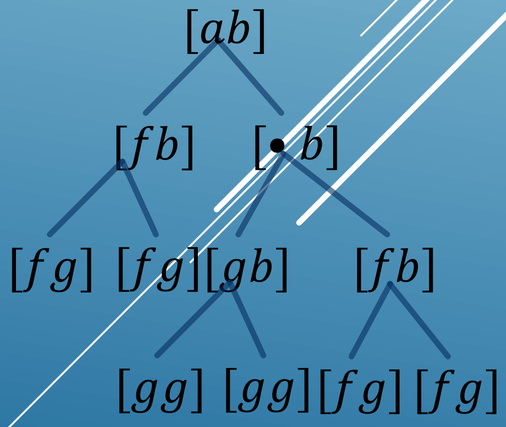
THE POLYNOMIAL REPRESENTATION

- $2fgx^2 + 2ggx^3 + 2fgx^3$ describes
  - all the elementary membranes created by all divisions
  - their multiset contents, and
  - the number of the corresponding computation steps
- Let's see:



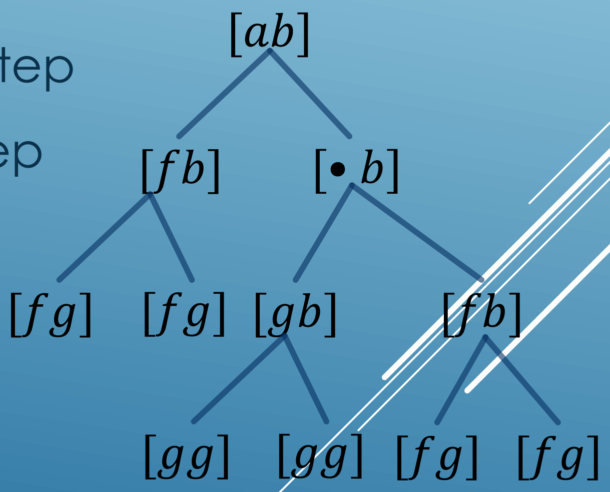
THE POLYNOMIAL REPRESENTATION

- We are not ready
  - We cannot calculate the product of all the odp's in polynomial time
  - But we can calculate efficiently the answer to the following question:
    - How many copies of an object are in those elementary membranes that are created at the nth step of the system and cannot be further divided?
- Consider the previous example and the object f
- Substitute 1 for each variable that is neither f nor x in
  - $(fx + gx^2 + fx^2) \cdot 2gx$
  - $(fx + x^2 + fx^2) \cdot 2x = 2fx^2 + 2x^3 + 2fx^3$
    - Two copies of f were created at the 2nd step
    - Two copies of f were created at the 3rd step



THE POLYNOMIAL REPRESENTATION

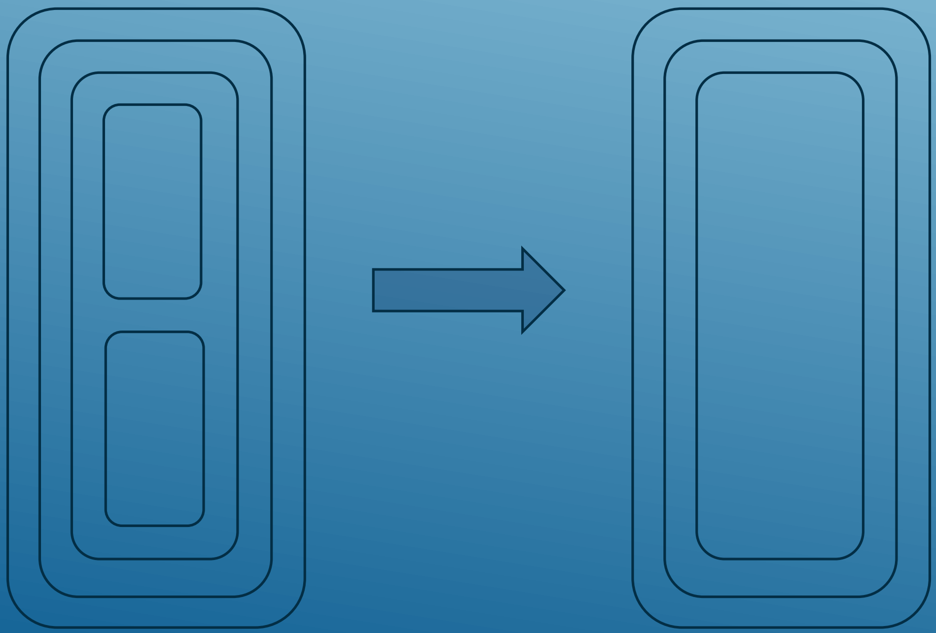
- Consider now the previous example and the object g
- Substitute 1 for each variable that is neither g nor x in
  - $(fx + gx^2 + fx^2) \cdot 2gx$
  - $(x + gx^2 + x^2) \cdot 2gx$
  - $= 2gx^2 + 2ggx^3 + 2gx^3$
  - $= 2gx^2 + (2g^2 + 2g)x^3$
  - Two copies of g were created at the 2nd step
  - Six copies of g were created at the 3rd step



- Can we really calculate this product efficiently in general?

SIMULATION IN POLYNOMIAL TIME

- Using this we can answer the following question now efficiently:
  - Which and how many objects are released to the parent membrane by the elementary membranes during a particular step of the system?
- We can build a polynomial time algorithm to simulate all membranes

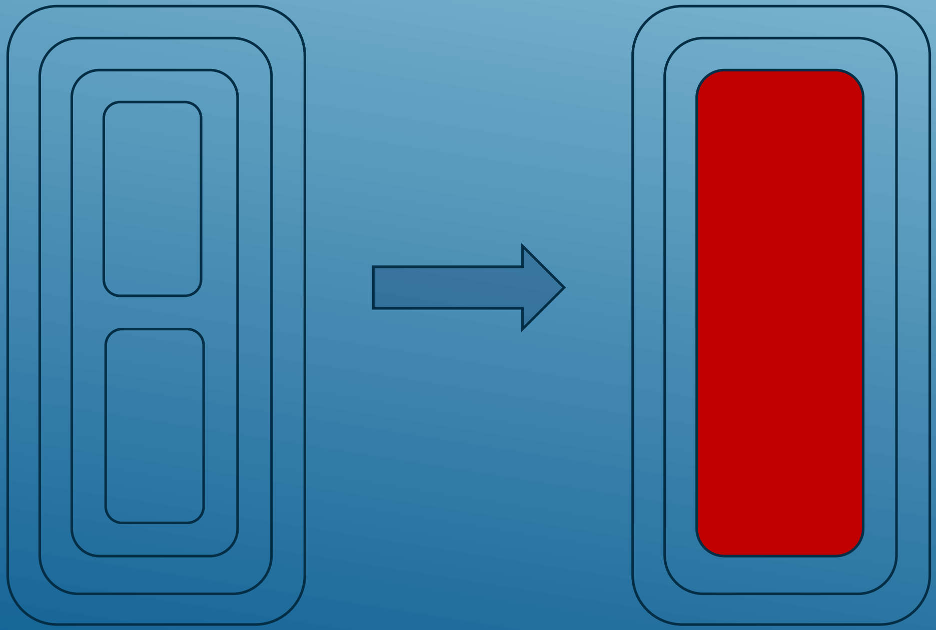


We can calculate efficiently:

- How many steps are needed to dissolve the all elementary membranes
- What objects get to the parent
- What happens with the rest of the system

SIMULATION IN POLYNOMIAL TIME

- Using this we can answer the following question now efficiently:
  - Which and how many objects are released to the parent membrane by the elementary membranes during a particular step of the system?
- We can build a polynomial time algorithm to simulate all membranes



Can we deal with that this membrane can contain <u>exponentially many</u> objects?

CONCLUSIONS

- What do we expect?
  - The answers to the red questions are positive
  - We can extend the method to out-communication rules
  - Maybe we can extend it to unit evolution rules (seems to be not so easy)
  - Maybe this method suits for implementations
- We have no idea yet how to extend this method to arbitrary polarizationless P systems