Parallel Biological *in silico* Simulation

P. Amar[1] M. Baillieul[2] D. Barth[2] B. LeCun[2]
F. Quessette[2] S. Vial[2]

[1]LRI, Paris-Sud University, FRANCE

[2]PRiSM, Versailles University, France

ANR MARMOTE ANR-12-MONU-0019

ISCIS 2014
Krakow, Poland
October 28, 2014

Outline

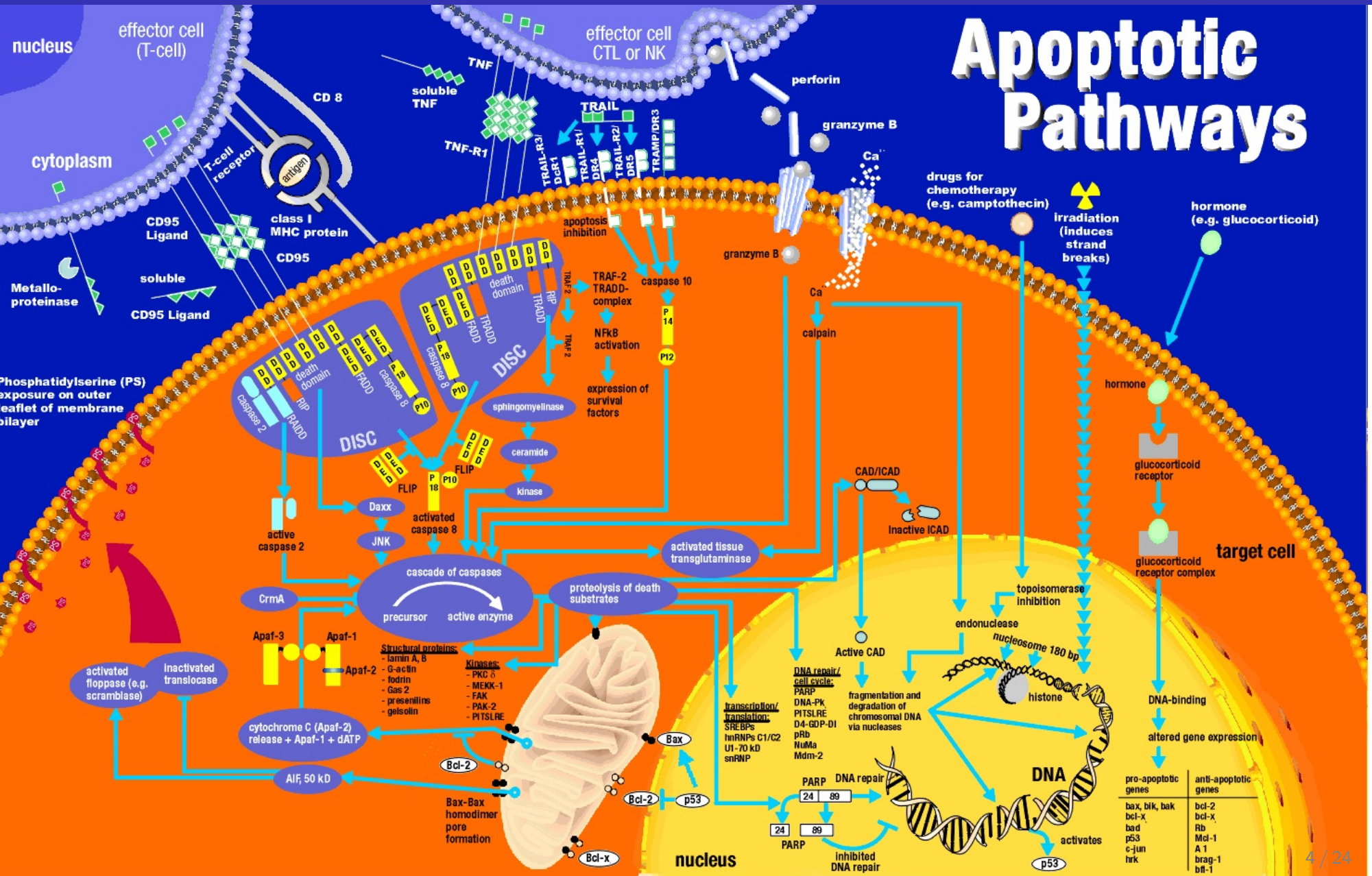
- Introduction: Apoptosis
- HSIM: Molecular Simulation
- BOBPP: Parallel Framework
- HSIM + BOBPP

Apoptosis

Objectives

- The metabolic and signaling pathways that lead to apoptosis (programmed death of the cell) are not very well understood for now.
- The different molecular compounds involved in apoptosis are well known.
- There are multiple apoptosis metabolic pathways, some of them seems to be quite independent from the others.
- All these pathways are not correctly activated and the apoptosis does not occur. This is typically what happens in some cancers.

Apoptosis


Apoptotic Pathways
nucleus
effector cell (T-cell)
effector cell CTL or NK
cytoplasm
CD 8
T-cell receptor
antigen
class I MHC protein
CD95 Ligand
CD95
soluble CD95 Ligand
Metallo-proteinase
TNF
soluble TNF
TNF-R1
TRAIL
TRAIL-R3 DcR1
TRAIL-R1 DR4
TRAIL-R2 DR5
TRAMP/DR3
perforin
granzyme B
Ca
drugs for chemotherapy (e.g. camptothecin)
irradiation (induces strand breaks)
hormone (e.g. glucocorticoid)
Phosphatidylserine (PS) exposure on outer leaflet of membrane bilayer
death domain
RIP
RAIDD
Caspase 2
FADD
Caspase 8
DISC
TRADD
TRAF 2
TRAF-2 TRADD-complex
caspase 10
apoptosis inhibition
NFkB activation
expression of survival factors
sphingomyelinase
ceramide
kinase
FLIP
Daxx
JNK
active caspase 2
activated caspase 8
cascade of caspases
precursor
active enzyme
CrmA
proteolysis of death substrates
activated tissue transglutaminase
calpain
CAD/ICAD
Inactive ICAD
Active CAD
topoisomerase inhibition
endonuclease
nucleosome 180 bp
histone
fragmentation and degradation of chromosomal DNA via nucleases
hormone
glucocorticoid receptor
glucocorticoid receptor complex
target cell
DNA-binding
altered gene expression
Apaf-3
Apaf-1
Apaf-2
Structural proteins:
- lamin A, B
- G-actin
- fodrin
- Gas 2
- presenilins
- gelsolin
Kinases:
- PKC δ
- MEKK-1
- FAK
- PAK-2
- PITSLRE
activated floppase (e.g. scramblase)
inactivated translocase
cytochrome C (Apaf-2) release + Apaf-1 + dATP
AIF, 50 kD
Bax
Bcl-2
Bax-Bax homodimer pore formation
Bcl-x
p53
DNA repair/ cell cycle:
PARP
DNA-Pk
PITSLRE
D4-GDP-DI
pRb
NuMa
Mdm-2
transcription/ translation:
SREBPs
hnRNPs C1/C2
U1-70 kD
snRNP
PARP
DNA repair
inhibited DNA repair
activates
DNA
nucleus
pro-apoptotic genes
bax, bik, bak
bcl-x
bad
p53
c-jun
hrk
anti-apoptotic genes
bcl-2
bcl-x
Rb
Mcl-1
A1
brag-1
bfl-1

Simulation

Context

- Very large number of acting molecules inside the cell involved in apoptosis (the simulation of a global cell is beyond computation power for now).
- The cell is naturally divided in organites, and each one may have a specific environment with its particularities.

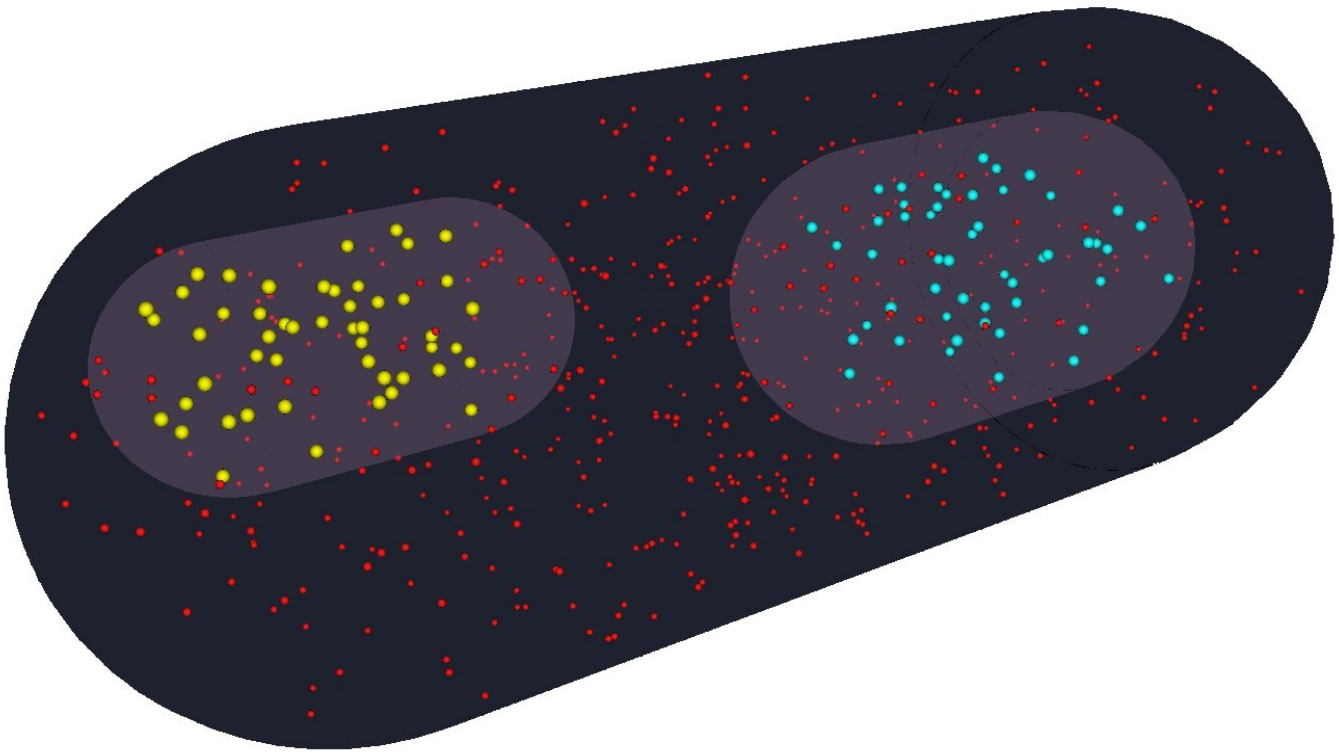
Tools

- HSIM is a cell simulator that allow to simulate only the interesting actors of the cell without ignoring the effects of the others.
- BOBPP is a parallel library that permits to simulate tasks in parallel. It is a high level tool that makes programming independent of the computer that runs the resulting program.

HSIM: the model

Entity centered model ...

- Each molecular species is a class, each molecule is an object.
- Environment: nested compartments surrounded by membranes.



HSIM: *small molecules*

Mixed with a global model

- Each compartment can contain a large number of copies of *small molecules* homogeneously distributed.
- Which can diffuse through the membranes (the diffusion speed is specific to each molecular species).
- They can interact with the other kind of molecules
- A variant of the Gillespie algorithm is used to manage the *small molecules*

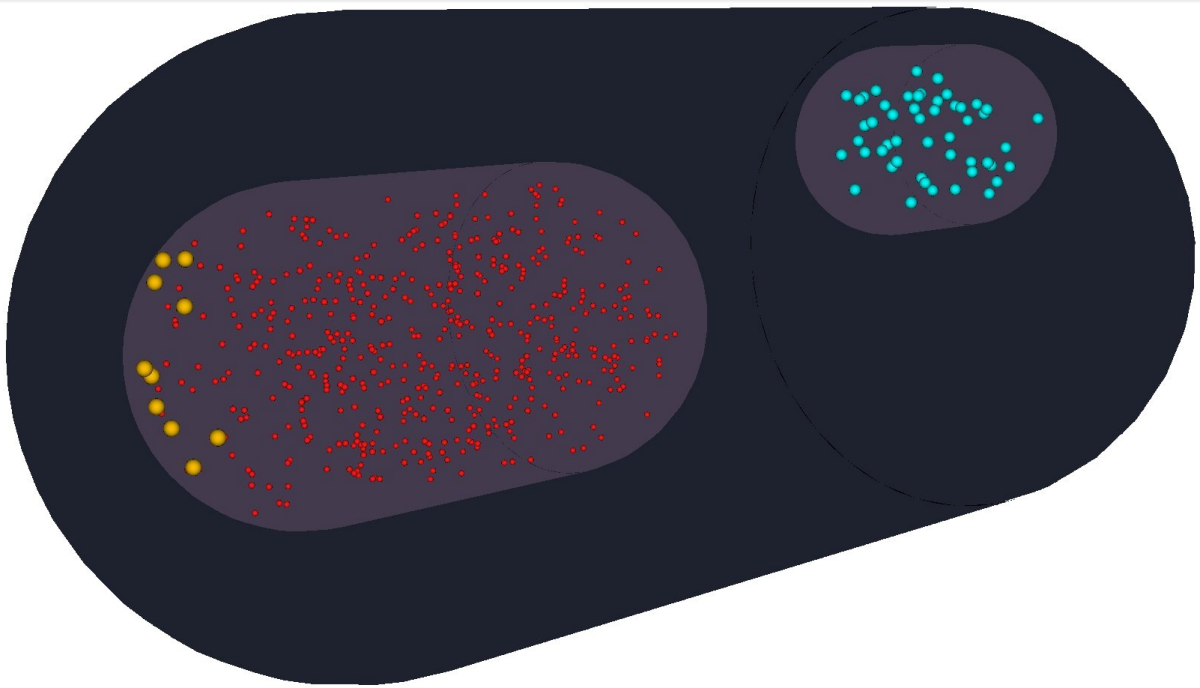
HSIM: the model

The diameter of the molecules varies between 1 and 10 nm

- The size ratio is from 1 to 1000.

Two classes of molecules

- Membrane bound which can diffuse in 2D in the membrane.
- The other ones diffuse in 3D inside the compartment.



HSIM: Evolution rules

The evolution is driven by rewriting rules mimicking biochemical reactions

- Allow to use theoretical tools (logical proofs, Petri nets, etc.).
- The model is written in a *ad hoc* language.
- Versatile tool (very different models can be simulated).

Output

- Real time 3D graphic display of the virtual cell.
- Curves showing the evolution of the concentrations of the reactants.
- The current state can be saved in a file for future use.
- Plotting curves with *excel* or *gnuplot*.
- Analysis of the spacial localisation of the molecules.

HSIM: Evolution rules

Basic rules:

Reaction	A + B	$\longrightarrow$	C + D	[Pr]
Association	A + B	$\longrightarrow$	C * D	[Pr]
Dissociation	A * B	$\longrightarrow$	C + D	[Pr]
Modification	A * B	$\longrightarrow$	C * D	[Pr]

Special cases:

Synthesis	A	$\longrightarrow$	A + B	[Pr]
Degradation	D + A	$\longrightarrow$	D	[Pr]
Renaming	A	$\longrightarrow$	B	[Pr]

HSIM: Evolution rules

Local evolution rules

- Not systematically applied, but according to a probability.
- Can be ambiguous (two rules may apply).

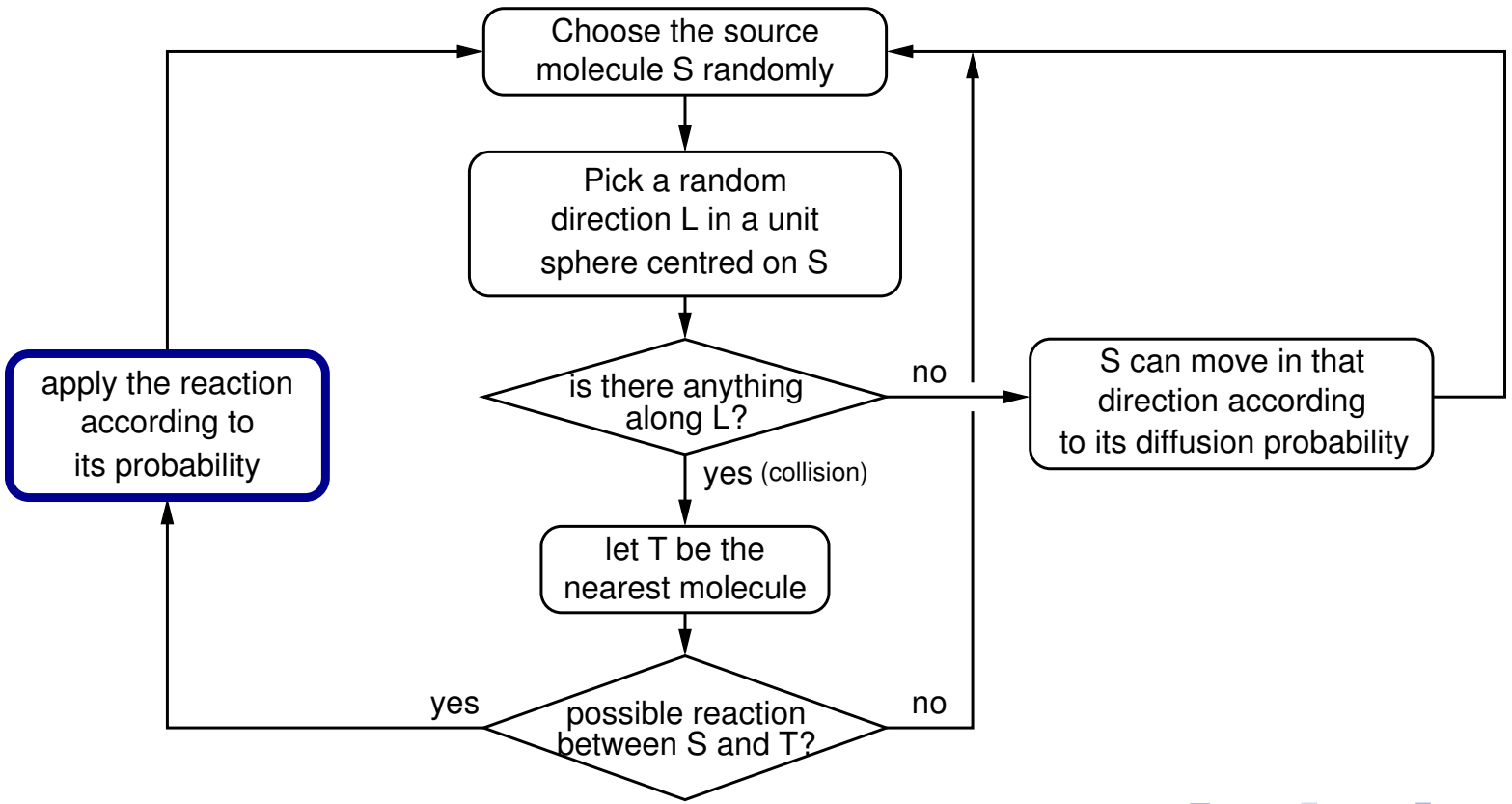
Non deterministic system

- Each simulation of the model use one trajectory.
- Many runs are necessary to get a statistically relevant result.

HSIM: Algorithm

One step of simulation

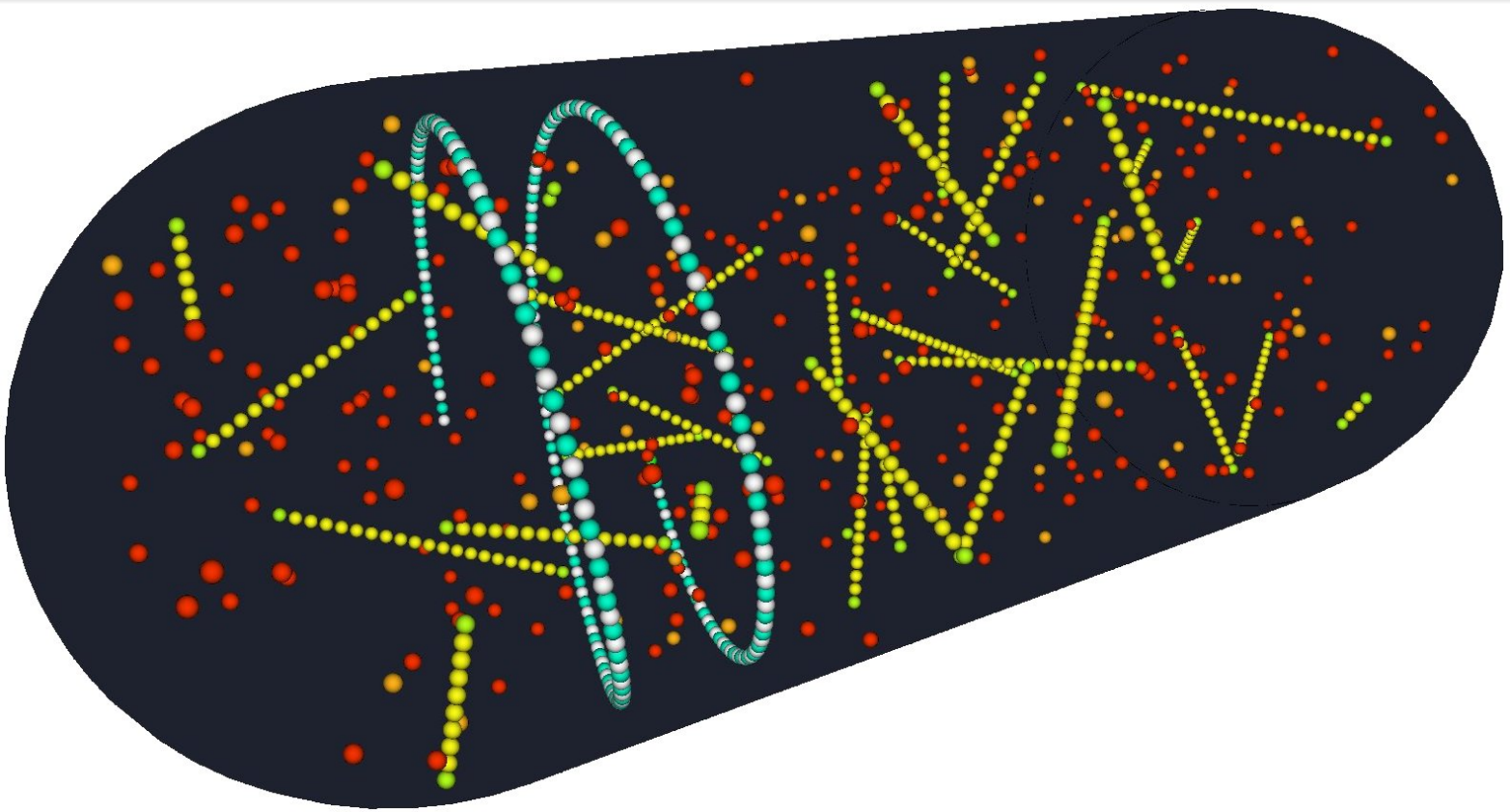
One step of simulation is finished when each molecule present at the current step has been processed according to the following algorithm:



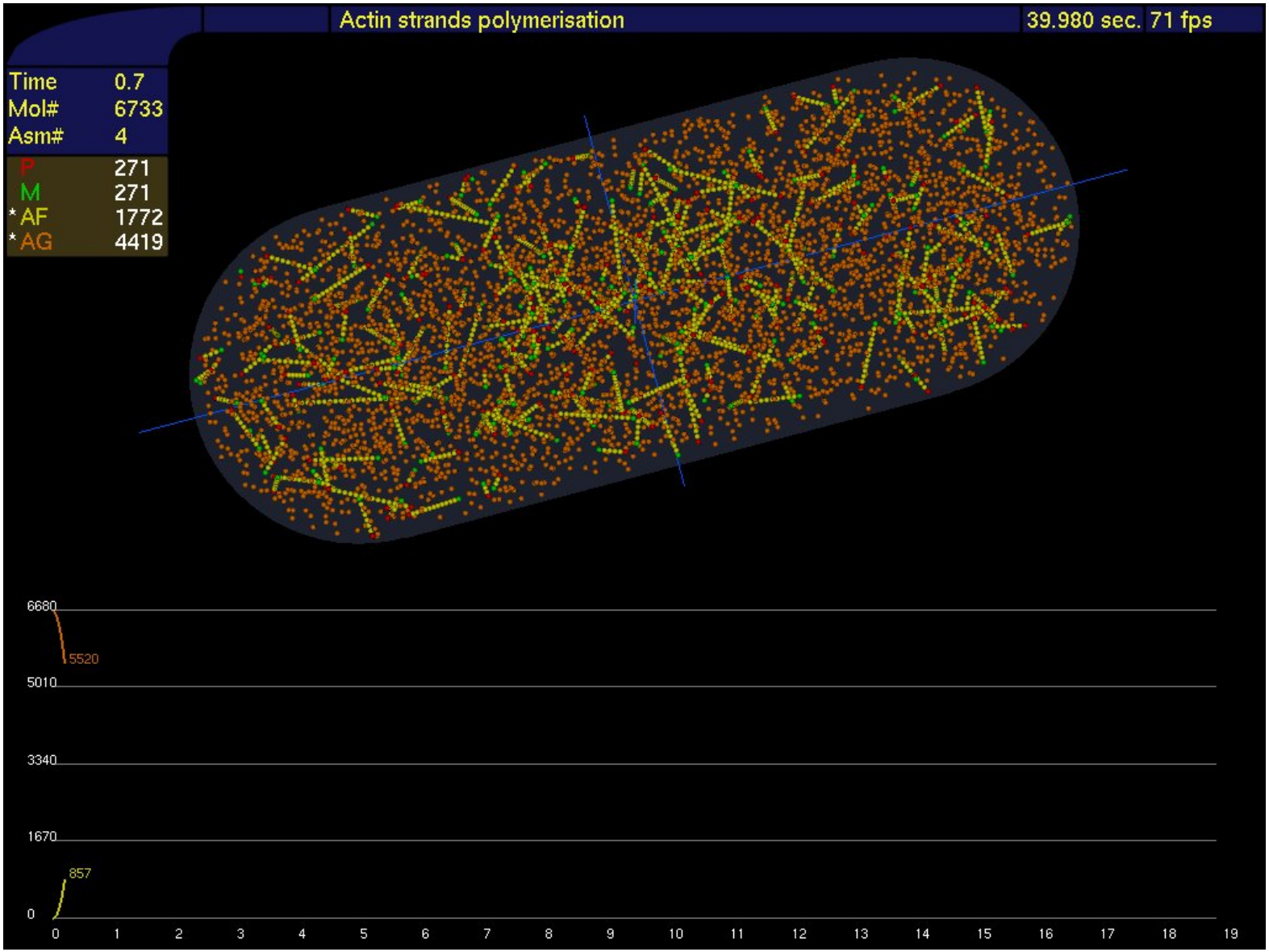
HSIM: Geometry of the assemblies

Possible geometries

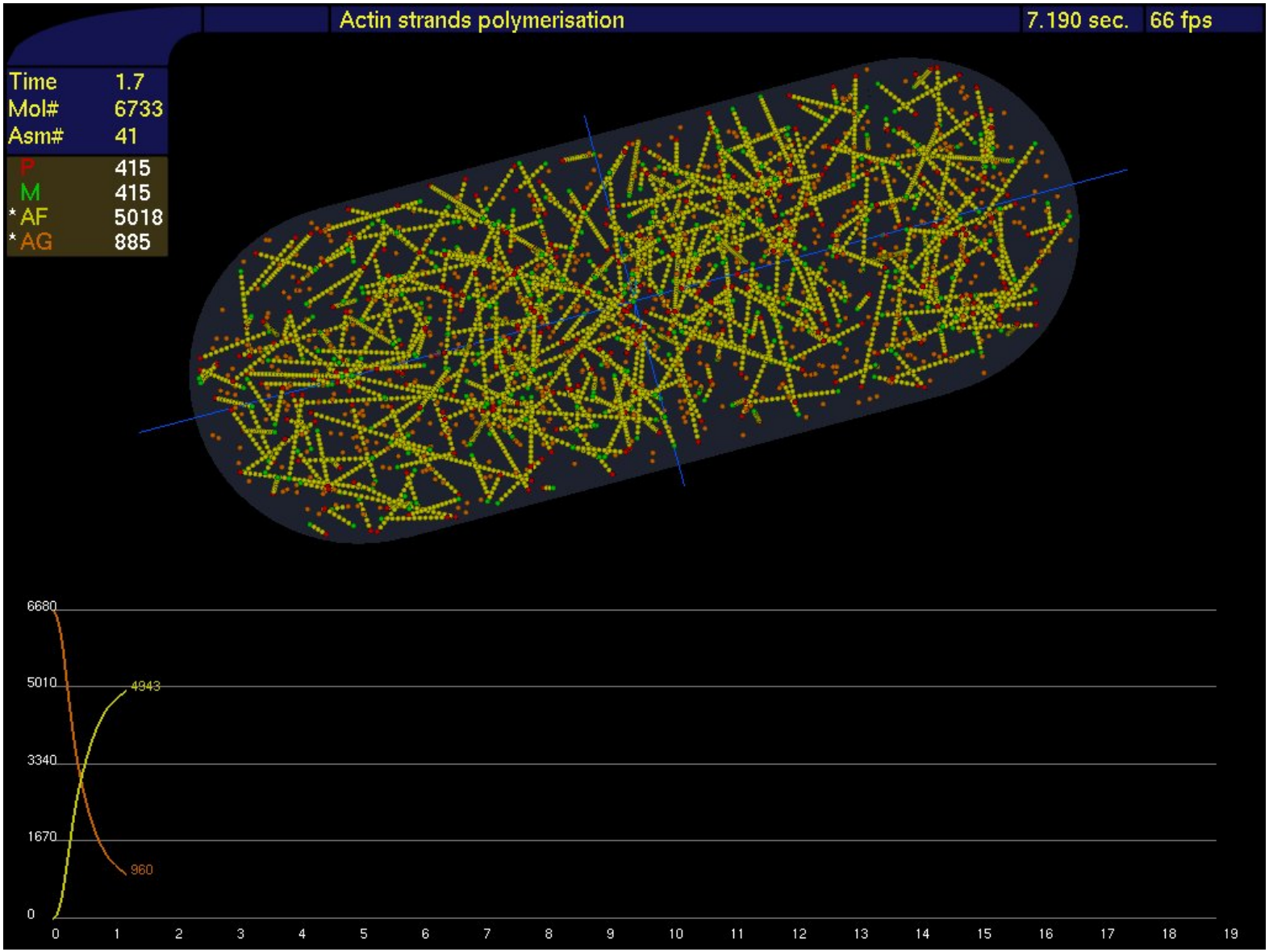
- Linear: $A + B \longrightarrow C = D$ [Pr]
- Helix: $A + B \longrightarrow C \ / \ D$ [Pr]



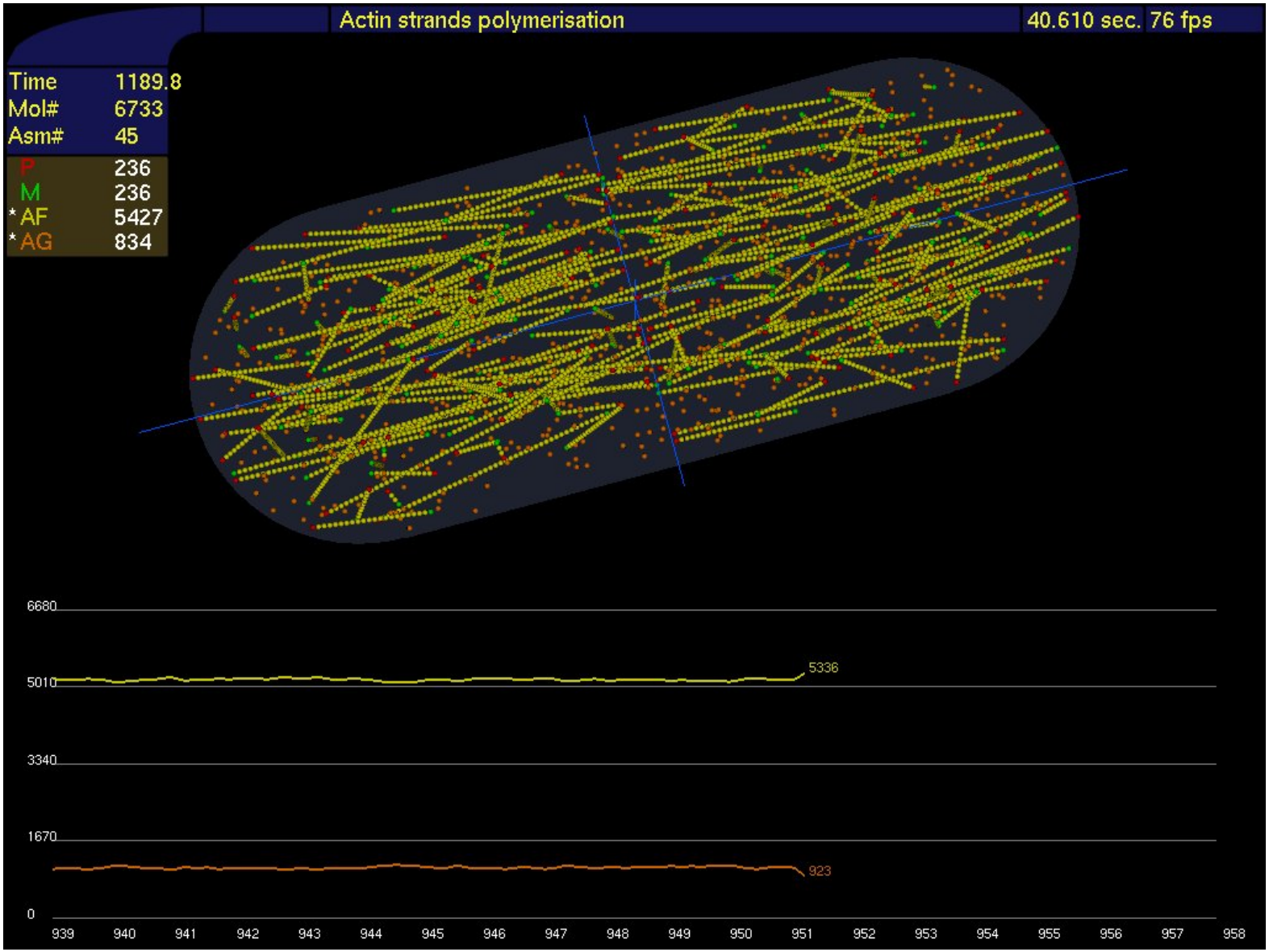
Example: polymerisation of actin strands 1/3



Example: polymerisation of actin strands 2/3



Example: polymerisation of actin strands 3/3



BOBPP: parallel Framework for Combinatorial Optimization

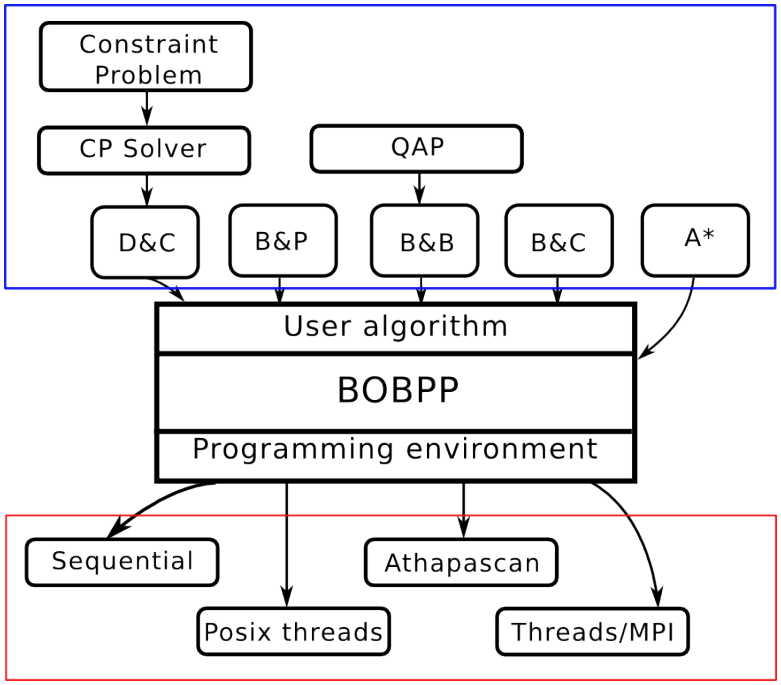
Objectives

- Solve Combinatorial Optimization Problems: Traveling Salesman problem, Quadratic assignment Problem, Golomb Ruler, ...
- Using search-tree methods: Divide & Conquer, Branch & Bound, ...
- On parallel machines: Shared memory: threads and mutex (pthreads), Distributed memory : processes and messages (MPI), Cluster of SMPs: threads and processes (pthreads+MPI).

BOBPP

Idea: Defining a programming model

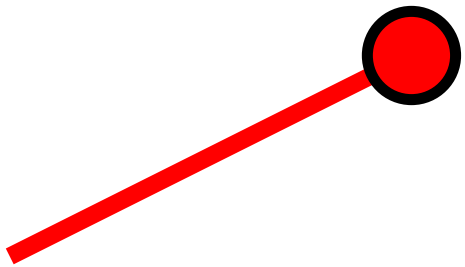
- Suitable for tree search algorithm.
- Suitable for parallel environments.



Tree Search algorihms

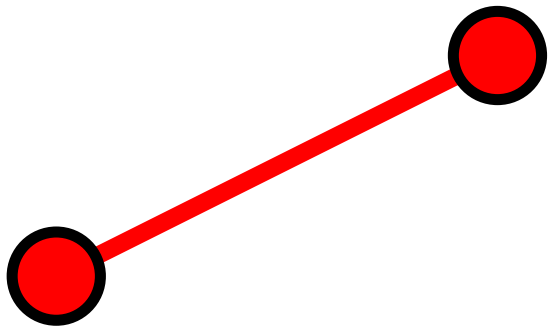
The user must **define** the problem, mainly the data stored in the node.

Tree Search algorihms



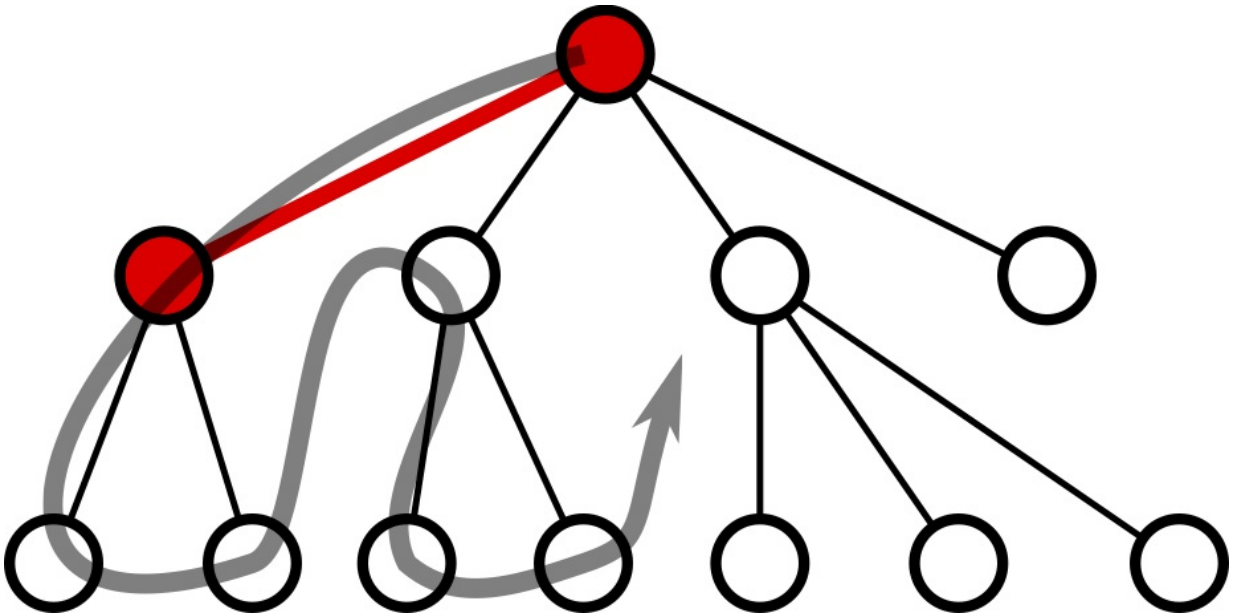
The user **describes** how a node is generated from a parent node (the child generation) according to the branching strategy.

Tree Search algorihms



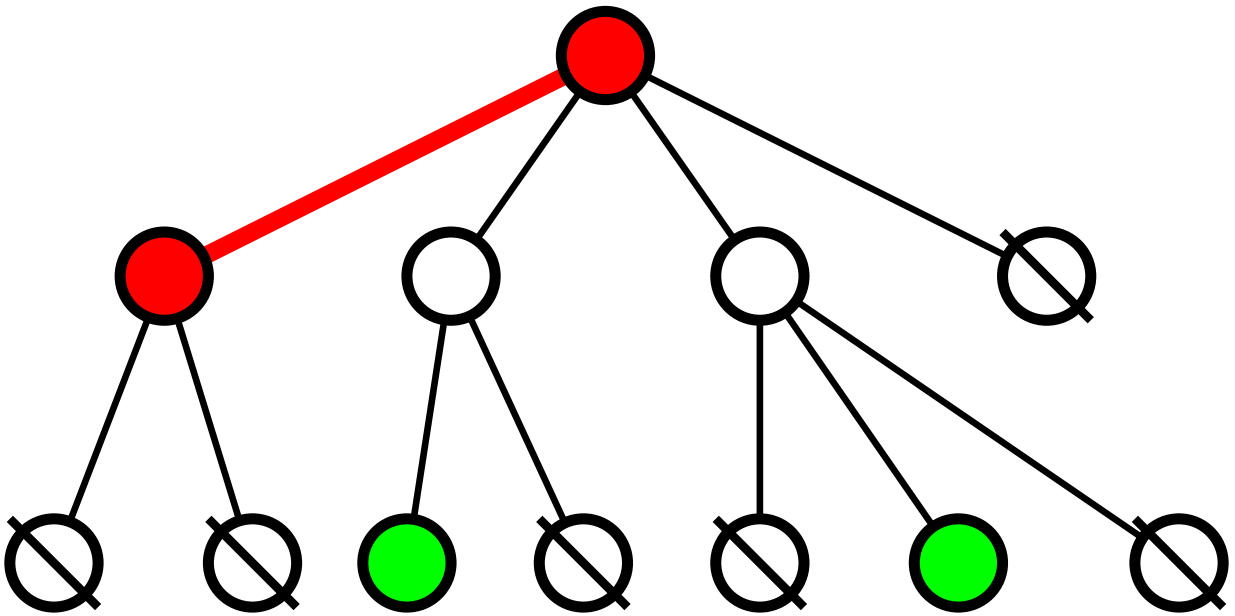
The user **describes** a work on node: evaluation function, constraint propagation, ...

Tree Search algorihms



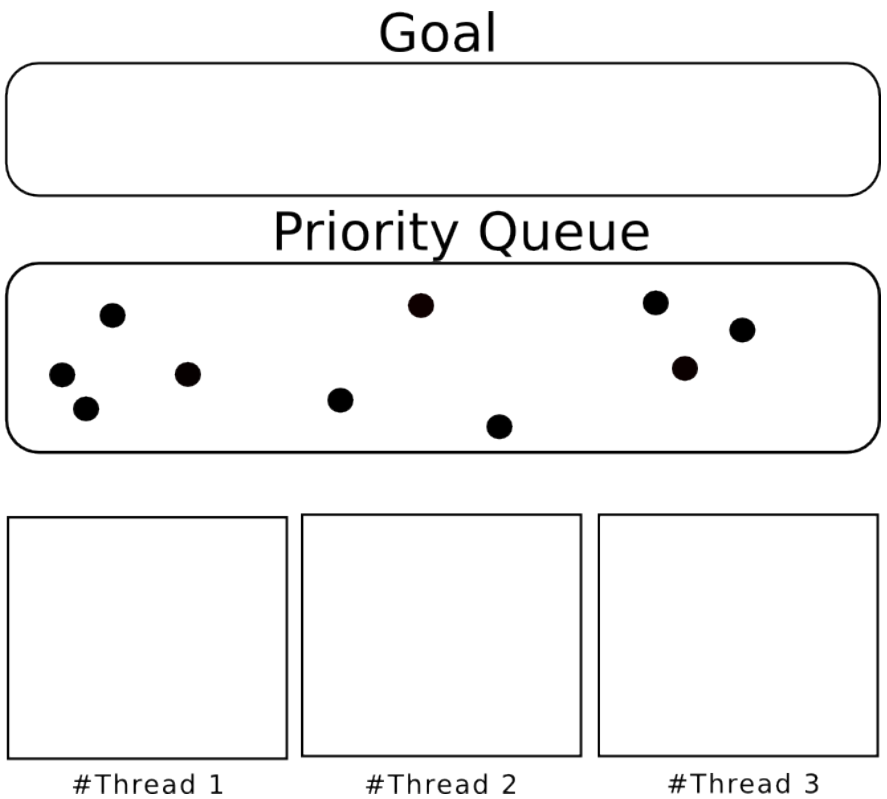
The user may **choose** between different exploration strategies (best first, depth-first search, etc)

Tree Search algorihms



The user may choose what is the goal of the search: the best solution, the number of feasible/best solutions $\Longrightarrow$ the stopping criteria.

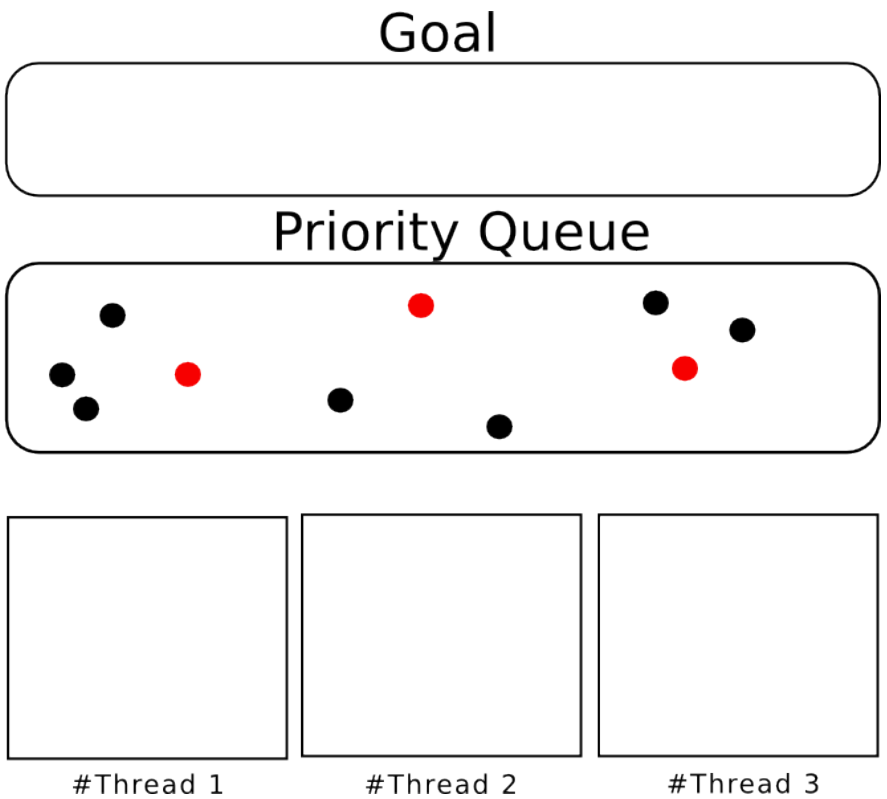
BOBPP and Parallelism



Steps

1. Nodes of the search-tree
2. Each thread selects one node
3. Each thread generates the children
4. If a solution is found, the goal is updated
5. The other nodes are inserted in the Priority Queue

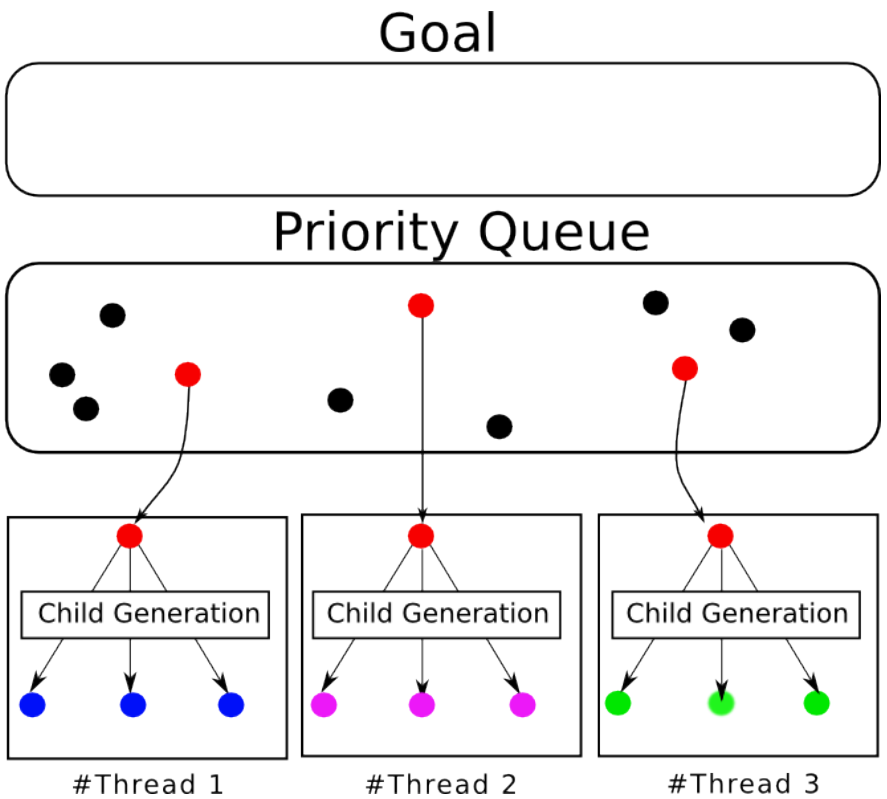
BOBPP and Parallelism



Steps

1. Nodes of the search-tree
2. Each thread selects one node
3. Each thread generates the children
4. If a solution is found, the goal is updated
5. The other nodes are inserted in the Priority Queue

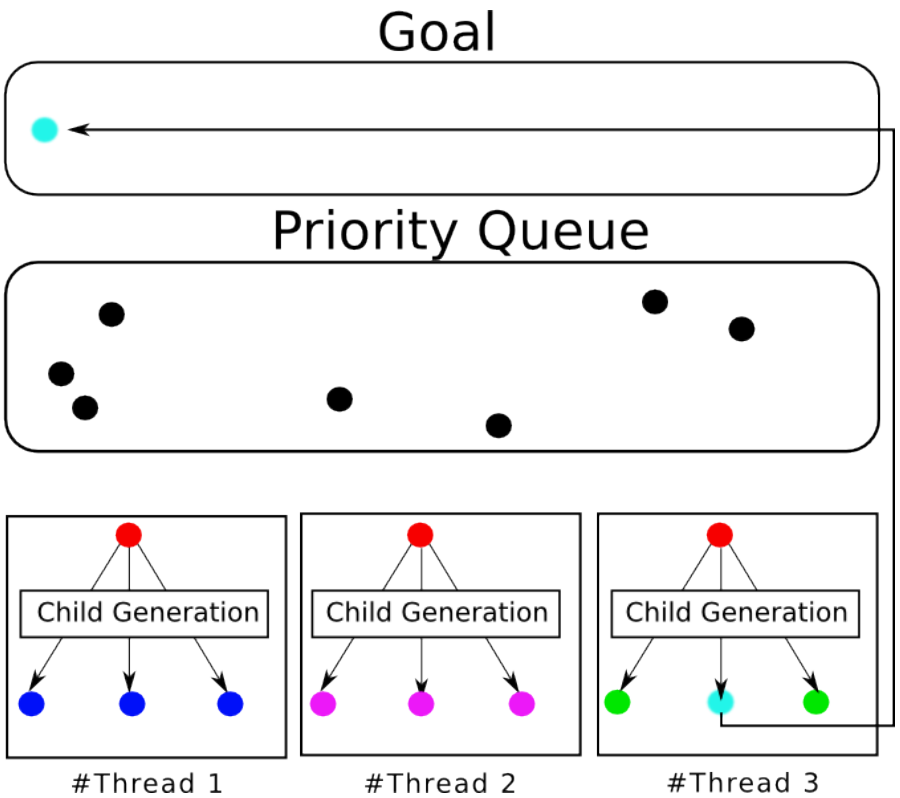
BOBPP and Parallelism



Steps

1. Nodes of the search-tree
2. Each thread selects one node
3. Each thread generates the children
4. If a solution is found, the goal is updated
5. The other nodes are inserted in the Priority Queue

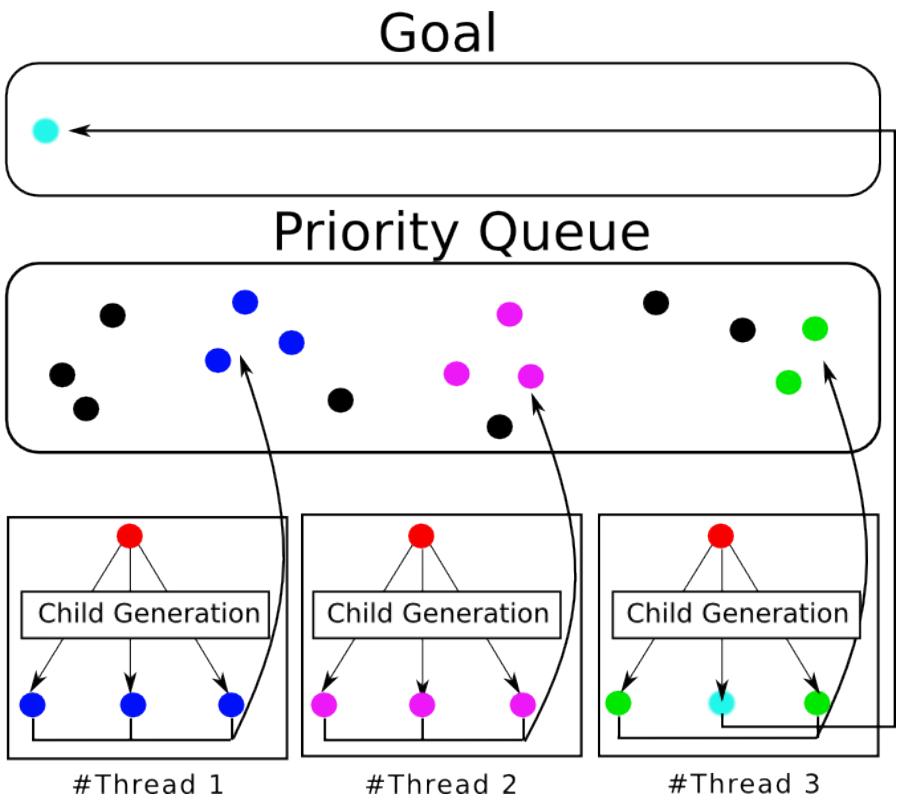
BOBPP and Parallelism



Steps

1. Nodes of the search-tree
2. Each thread selects one node
3. Each thread generates the children
4. If a solution is found, the goal is updated
5. The other nodes are inserted in the Priority Queue

BOBPP and Parallelism



Steps

1. Nodes of the search-tree
2. Each thread selects one node
3. Each thread generates the children
4. If a solution is found, the goal is updated
5. The other nodes are inserted in the Priority Queue

BOBPP and HSIM 1/5

Simulation in a Space

- A node: represents a voxel.
- At each step, node are not generated but modified.
- Communication between nodes
  - Adding links between nodes to enable the communication.
  - A node can be executed at the iteration n if it has received all message from iteration $n-1$.
  - Node may be ready or not ready (waiting state).
- Synchronization is only local.
- No need of Global Barrier.
- Hypothesis: many particles are communicated between nodes...
- More nodes than cores to balance the load.

BOBPP and HSIM 2/5

Links

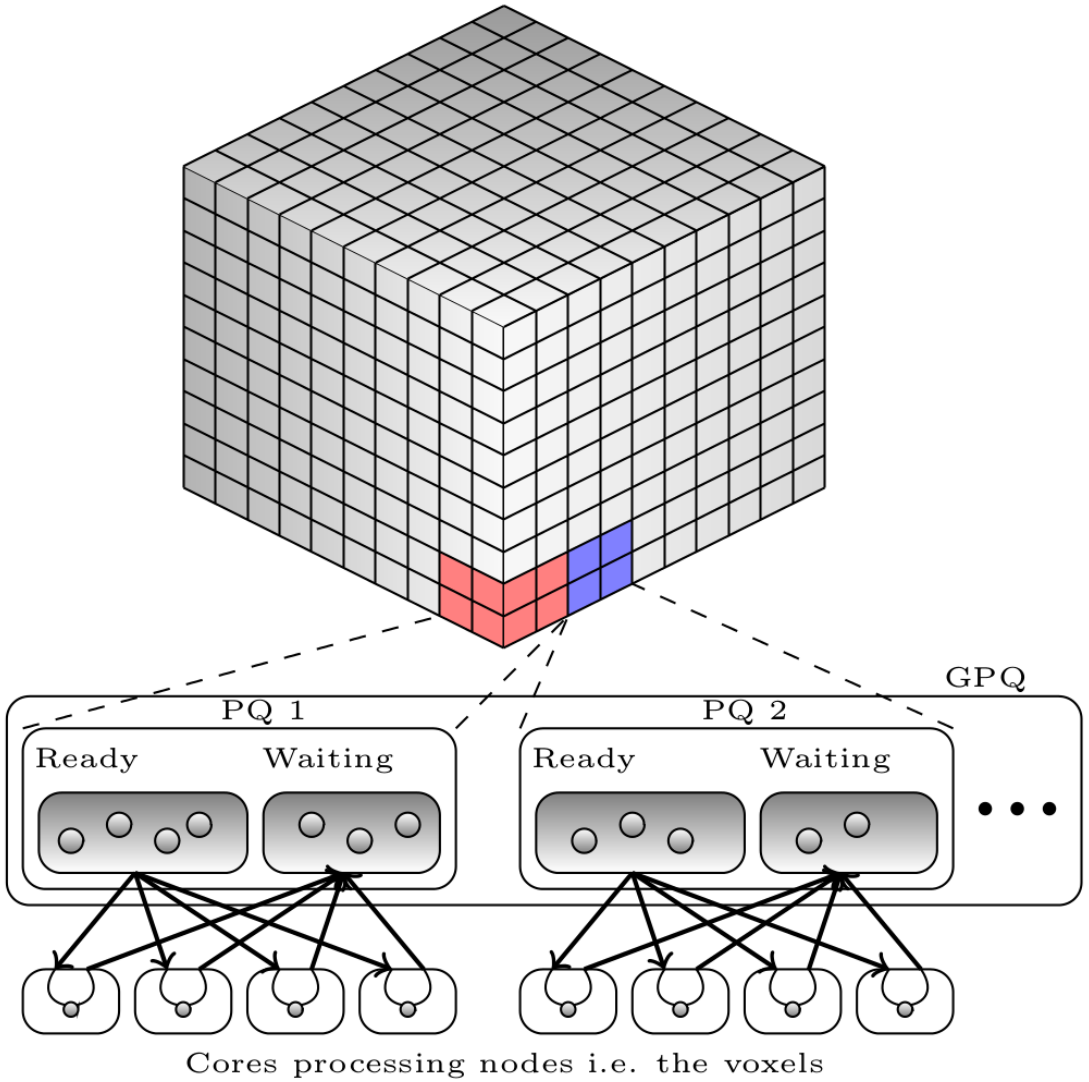
- We add a specific bi-directional communication canal called a `Link`. Must be declared between two nodes that exchange particles. For a 3-dimensional space, each node must have six links.
- A node is ready for the iteration n if it has message from all its links at the iteration $n - 1$.
- At the beginning of an iteration, a node gets all the messages from its `Links` and at the end of its iteration it sends messages on all its links.

BOBPP and HSIM 3/5

Priority queue

- We add a specific implementation of priority queue which is composed by two queues to handle the two node's states: waiting and ready.
- The first one the waiting queue stores the waiting nodes, although the second one called the ready queue stores the ready nodes.
- When a node become ready, it is deleted from the waiting queue and is inserted in the ready queue.

BOBPP and HSIM 4/5



BOBPP and HSIM 5/5

Advantages of the approach

- This implementation is asynchronous, the mechanism of `Link` with tagged messages ensures that a node has all the necessary information to complete the execution at the iteration n.
- The synchronicity is only local, between pair of nodes.
- There is no need of time consuming mechanism like global barrier or global mechanism to be sure that the execution is correct.
- This local synchronous mechanism ensures a correct global execution of the system.
- The partitioning of the global space in more parts than the number of cores, permits a good load balancing.
- A fixed number of cores is associated to a a priority queue and each queue can store many more nodes that the number of computing cores. While one core executes a long task, another core can executed several small tasks.

Software free download

HSIM

`http://www.lri.fr/~pa/Hsim`

BOBPP

`https://forge.prism.uvsq.fr/projects/bobpp`

Small Demo