The Combinatorics of Barrier Synchronization

presented last week at Petri Nets'19

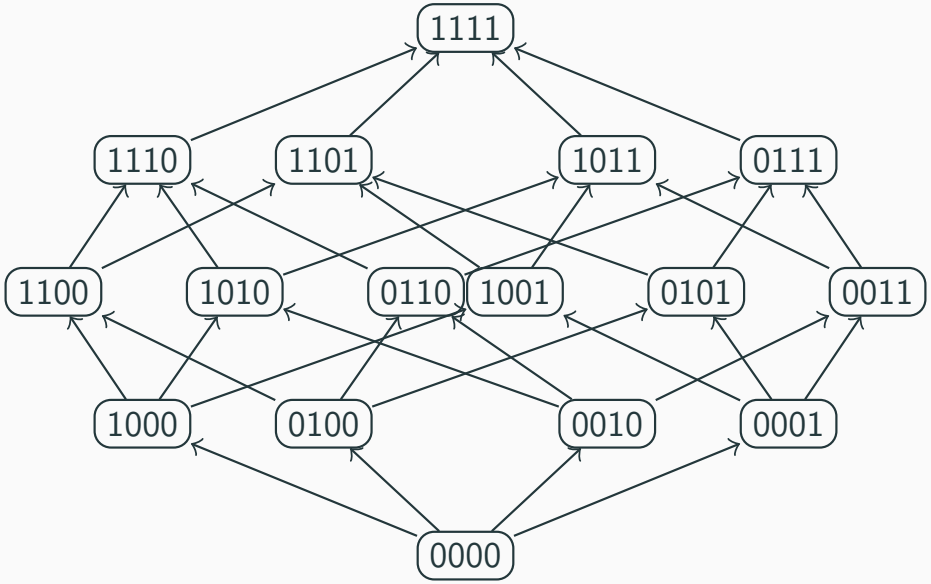
Olivier Bodini, Matthieu Dien,
Antoine Genitrini, Frédéric Peschanski,

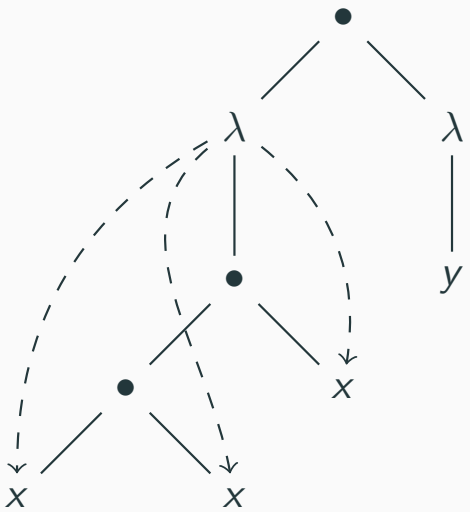
July 2nd 2019

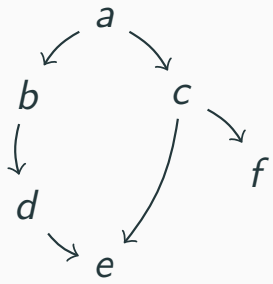
CLA'19

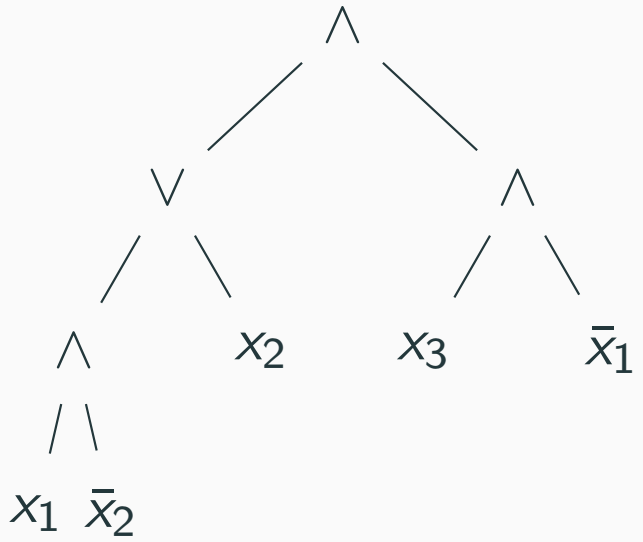
Syntactic domains

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg F, F, \neg G \vdash F}\,Ax \quad \dfrac{}{\neg F, F, \neg G \vdash \neg F}\,Ax}{\neg F, F, \neg G \vdash \bot}\,\neg e}{\neg F, F \vdash G}\,Abs}{\neg F \vdash F \to G}\,{\to}i}{\vdash \neg F \to (F \to G)}\,{\to}i$$





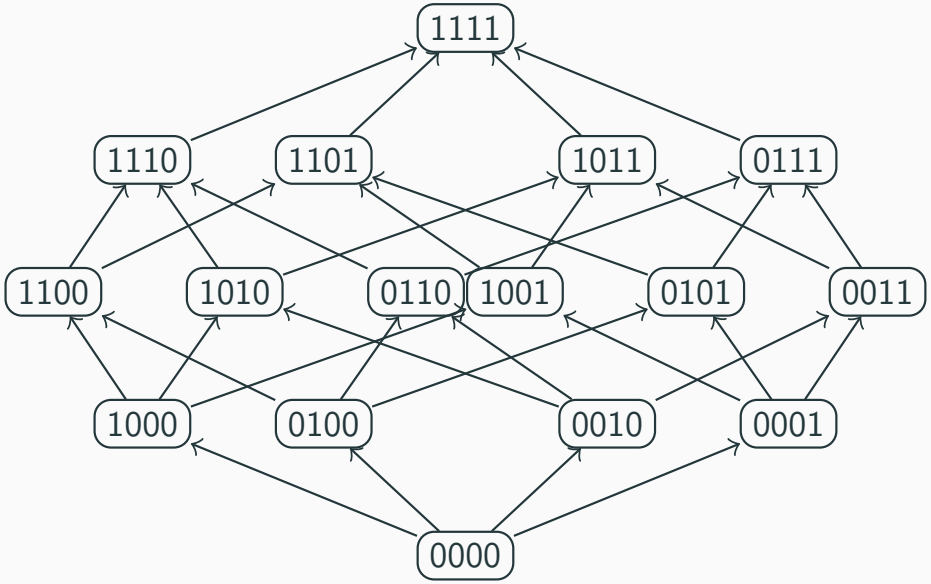


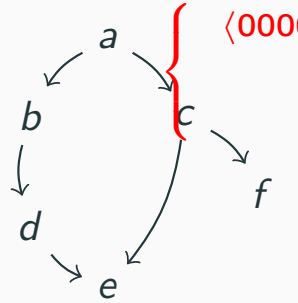


Syntactic domains and Semantic domains

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg F, F, \neg G \vdash F}\,Ax \quad \dfrac{}{\neg F, F, \neg G \vdash \neg F}\,Ax}{\neg F, F, \neg G \vdash \bot}\,\neg e}{\neg F, F \vdash G}\,Abs}{\neg F \vdash F \to G}\,\to i}{\vdash \neg F \to (F \to G)}\,\to i$$

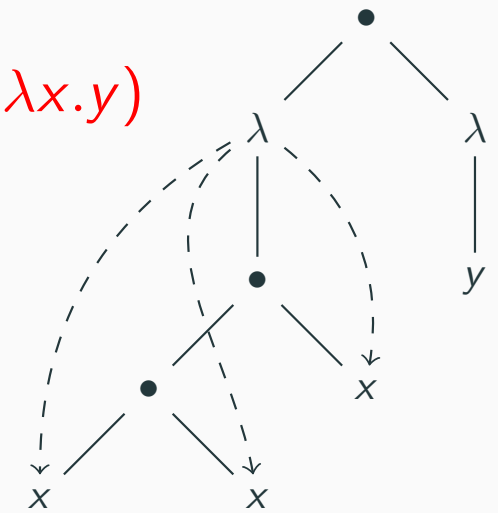
$\vdash \neg F \to (F \to G)$



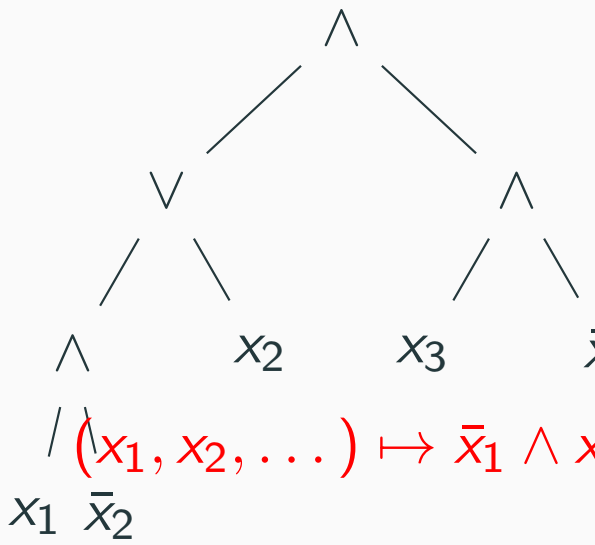


$\{\ \langle 0000, 1000, 0100, 0010, 0001, 1100, 1010, 0110, 1001, 0101, 0011, 1110, 1101, 1011, 0111, 1111\rangle \ \ldots\ \}$

$y(\lambda x.y)$



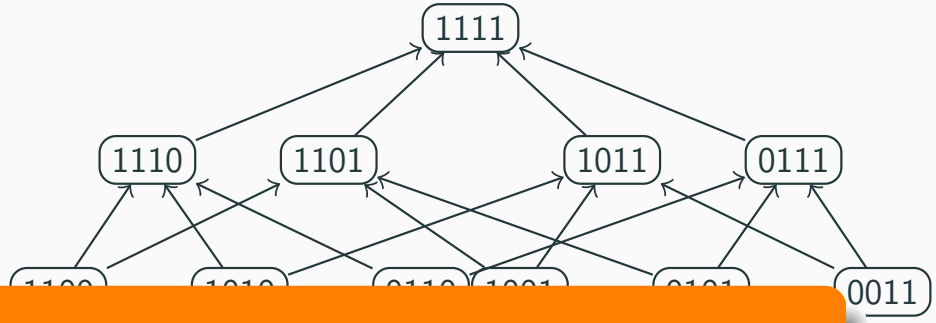
$\{\ \langle a, b, d, c, e, f\rangle\ \langle a, b, c, d, f, e\rangle\ \ldots\ \}$



$(x_1, x_2, \ldots) \mapsto \bar{x}_1 \wedge x_2 \wedge x_3$

Syntactic domains and Semantic domains

$$\dfrac{\dfrac{\dfrac{}{\neg F, F, \neg G \vdash F}\,Ax \quad \dfrac{}{\neg F, F, \neg G \vdash \neg F}\,Ax}{\neg F, F, \neg G \vdash \bot}\,\neg e}{\neg F, F \vdash G}\,Abs$$

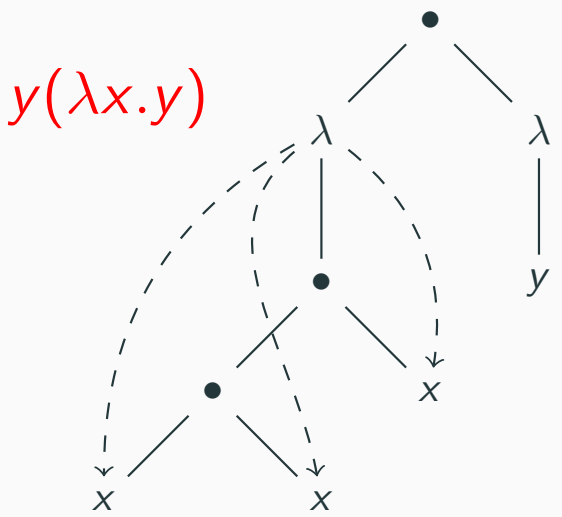


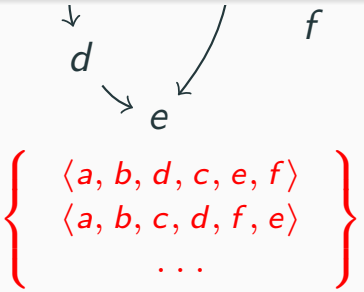
Outline

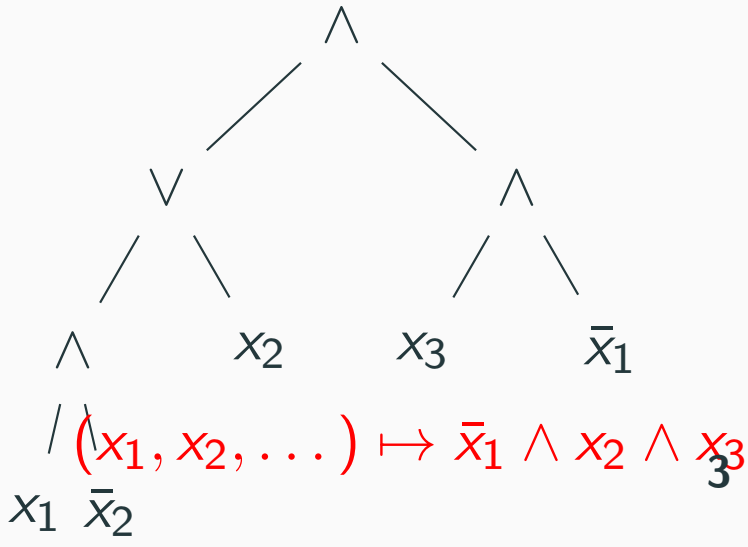
1. Concurrency theory context
2. Algorithm tools
3. Complexity results

$\vdash \neg$

01,
1⟩







Concurrency theory context

Object of study

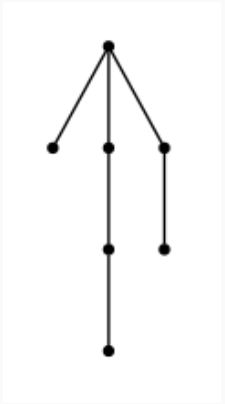
Concurrency Theory

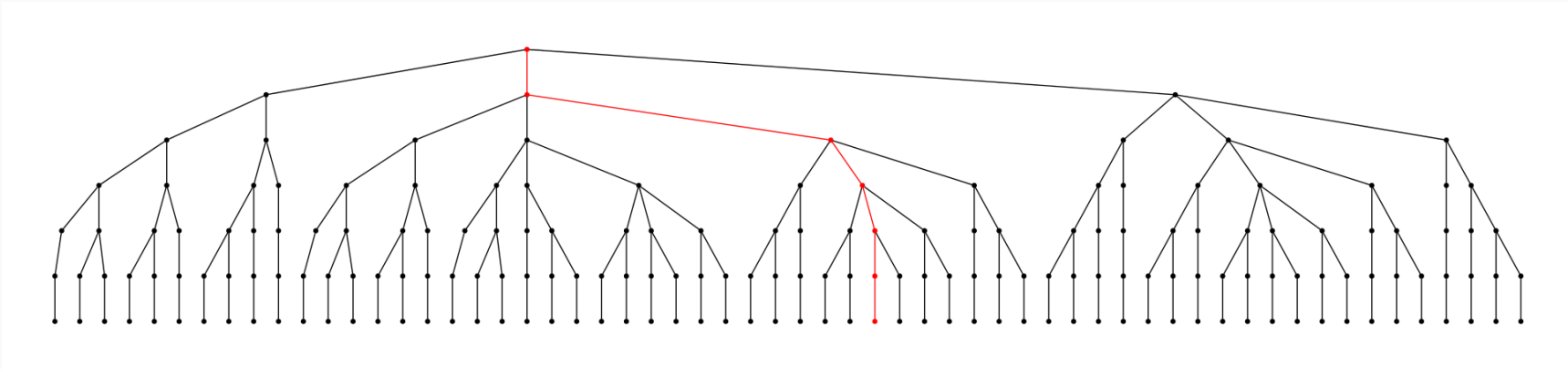
Languages and formalisms defining concurrency:
parallelism, synchronization, non-determinism, deadlocks,. . .

Object under study: Simplified Process Algebra

Main source of explosion

When analyzing concurrent processes, the parallel operator is the main source of ***combinatorial explosion****.* [Mi80], [ClGrPe99]





Object of study

Concurrency Theory

Languages and formalisms defining concurrency:
parallelism, synchronization, non-determinism, deadlocks,. . .

Object under study: Simplified Process Algebra
and its classical combinatorial explosion phenomenon.

Our point of view

Understanding the combinatorics that interacts in this combinatorial explosion:

- Poset theory
- Enumerative combinatorics
- Analytic combinatorics

Practical applications?

Question: is there any practical use of *this*?

$\Rightarrow$ we would argue yes, and it is about the *statistical* analysis of (concurrent) systems.

- generate executions uniformly at random
- "navigate" wrt. the uniform distribution,
  e.g. exploring "less probable" parts of the system under study
  (skewing the uniform distribution)
- statistical model-checking[1]

[1] cf. *Monte Carlo model checking*, R. Gosu and S. A. Smola, Tacas 2005.

A simple class: barrier synchronization processes

A very simple calculus of *barrier synchronization*.

Process

$$
\begin{array}{lll}
P, Q ::= & 0 & \text{(termination)} \\
& \mid \alpha.P & \text{(atomic action and prefixing)} \\
& \mid \nu(B)P & \text{(barrier and scope)} \\
& \mid \langle B \rangle P & \text{(synchronization)} \\
& \mid P \parallel Q & \text{(parallel)}
\end{array}
$$

$$0,\ \alpha.0,\ \langle B \rangle\ 0,\ \nu(B)\ 0,\ \alpha.\beta.0,\ \ldots,$$

$$\nu(B)\ \ [a_1.\langle B \rangle a_2.0 \parallel \langle B \rangle b_1.0 \parallel \langle B \rangle 0],\ \ \ldots$$

A simple class: barrier synchronization processes

A very simple calculus of *barrier synchronization*.

Process		**Size** $\lvert . \rvert$	
$P, Q ::=$	0	0	(termination)
	$\mid \alpha.P$	$1 + \lvert P \rvert$	(atomic action and prefixing)
	$\mid \nu(B)P$	$\lvert P \rvert$	(barrier and scope)
	$\mid \langle B \rangle P$	$\lvert P \rvert$	(synchronization)
	$\mid P \parallel Q$	$\lvert P \rvert + \lvert Q \rvert$	(parallel)

Remark: ✓combinatorial class

$$0,\ \alpha.0,\ \langle B \rangle\ 0,\ \nu(B)\ 0,\ \alpha.\beta.0,\ \ldots,$$

$$\nu(B)\ [a_1.\langle B \rangle a_2.0 \parallel \langle B \rangle b_1.0 \parallel \langle B \rangle 0],\ \ldots$$

A simple class: barrier synchronization processes

A very simple calculus of *barrier synchronization*.

Process		**Size** $\lvert . \rvert$	
$P, Q ::=$	0	0	(termination)
	$\mid \alpha.P$	$1 + \lvert P \rvert$	(atomic action and prefixing)
	$\mid \nu(B)P$	$\lvert P \rvert$	(barrier and scope)
	$\mid \langle B \rangle P$	$\lvert P \rvert$	(synchronization)
	$\mid P \parallel Q$	$\lvert P \rvert + \lvert Q \rvert$	(parallel)

Remark: ✓combinatorial class

$$0,\ \alpha.0,\ \langle B \rangle\ 0,\ \nu(B)\ 0,\ \alpha.\beta.0,\ \ldots,$$
$$\nu(B)\ \ [a_1.\langle B \rangle a_2.0 \parallel \langle B \rangle b_1.0 \parallel \langle B \rangle 0],\ \ \ldots$$

$\Rightarrow$ what about a **semantic** notion of a size?

Semantics

Process behavior in a nutshell

$P \stackrel{\text{def}}{=} \nu(B)\ [a_1.\langle B\rangle a_2.0 \parallel \langle B\rangle b_1.0 \parallel \langle B\rangle 0]$

$\Rightarrow$ synchronization on barrier $\langle B\rangle$ is *not* available

$\cdots \xrightarrow{a_1} \nu(B)\ [\langle B\rangle a_2.0 \parallel \langle B\rangle b_1.0 \parallel \langle B\rangle 0]$

$\Rightarrow$ synchronization available

$\cdots \rightarrow a_2.0 \parallel b_1.0 \parallel 0$

$\Rightarrow$ interleaving semantics

$\cdots \xrightarrow{a_2}\xrightarrow{b_1} 0$ or $\cdots \xrightarrow{b_1}\xrightarrow{a_2} 0$

Definition (**Execution**) A maximal path of transitions

2 paths: $P \xrightarrow{a_1}\rightarrow\xrightarrow{a_2}\xrightarrow{b_1} 0$ and $P \xrightarrow{a_1}\rightarrow\xrightarrow{b_1}\xrightarrow{a_2} 0$

$\Rightarrow$ (semantic) size 2

Counting executions?

Why taking the number of (interleaved) executions as size?

▷ **Intuitively** it lets us observe/reason about *combinatorial explosion*.

▷ **More concretely** it is a very effective *enumerative combinatorics* tool by discriminating process terms in a very sharp way.

e.g. $\alpha_1.\alpha_2.\alpha_3.\alpha_4.0$ has syntaxic size 4 and semantic size 1

whereas $\alpha_1.\alpha_2.0 \parallel \alpha_3.\alpha_4.0$ has syntaxic size 4 and semantic size 6

Counting in general is hard

non trivial A *non-deadlocked* process expressed in the *barrier synchronization calculus* has a control graph shaped after an *intransitive directed acyclic graph* (DAG)

easy The correspondence is complete: any (intransitive) DAG can be expressed as a process

- The correspondence conveys to *partially ordered sets* (Posets)

Counting in general is hard

non trivial A *non-deadlocked* process expressed in the *barrier synchronization calculus* has a control graph shaped after an *intransitive directed acyclic graph* (DAG)

easy The correspondence is complete: any (intransitive) DAG can be expressed as a process

- The correspondence conveys to *partially ordered sets* (Posets)

Consequences:

- Process executions = Linear extensions
- Counting process executions = Counting linear extensions
- Counting process executions is ♯-P complete

Counting Linear Extensions is ♯-P complete, Brightwell and Winkler, 1991.

Counting in general is hard

non trivial A *non-deadlocked* process expressed in the *barrier synchronization calculus* has a control graph shaped after an *intransitive directed acyclic graph* (DAG)

easy The correspondence is complete: any (intransitive) DAG can be expressed as a process

- The correspondence conveys to *partially ordered sets* (Posets)

Consequences:

- Process executions = Linear extensions
- Counting process executions = Counting linear extensions
- Counting process executions is ♯-P complete

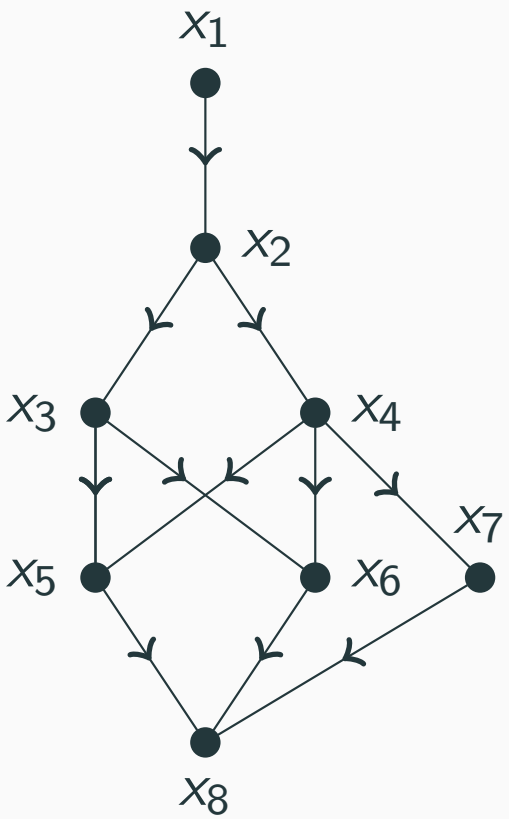
Counting Linear Extensions is ♯-P complete, Brightwell and Winkler, 1991.

. . . However there is a uniform random sampler available (Fast perfect sampling of linear extensions, Huber, 2006).

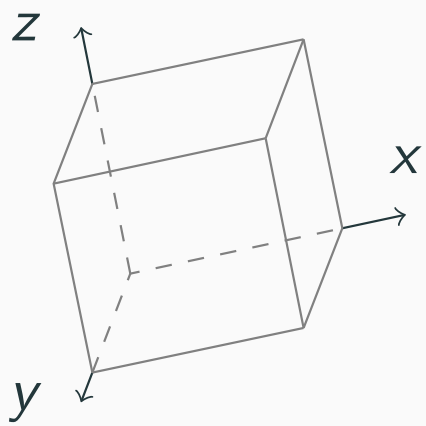
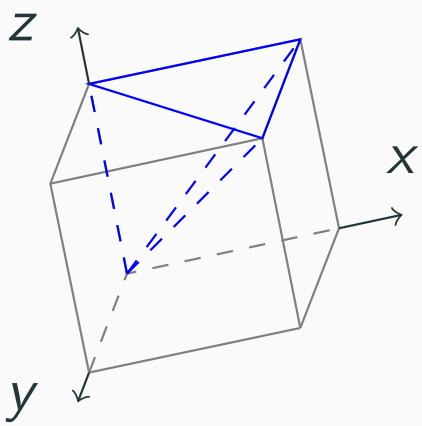
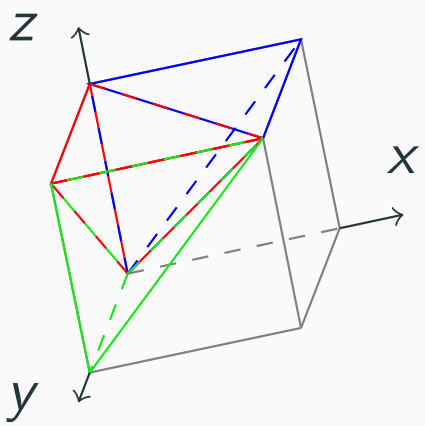
Algorithmic tools

Example

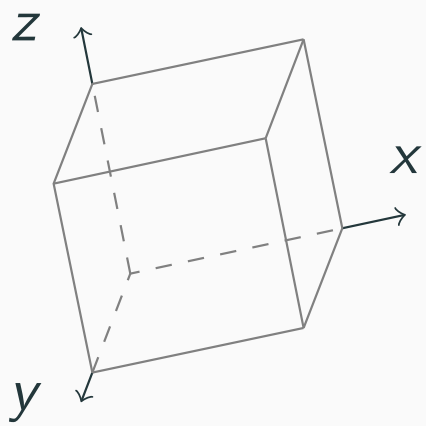
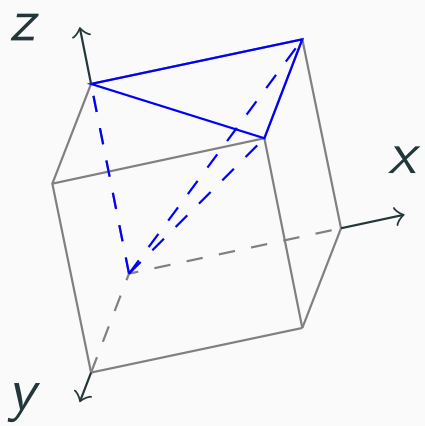
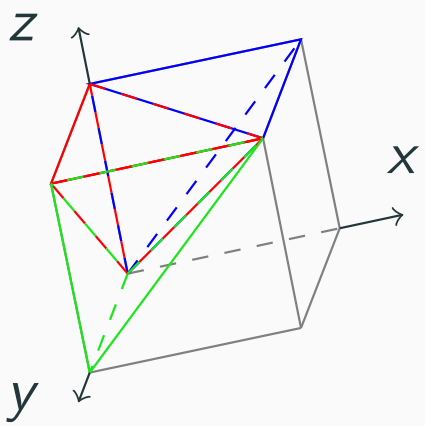
Question: Counting and uniformly random sampling of executions?



Order polytope [Stanley'86]

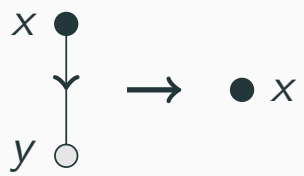
partial order	$x \quad y \quad z$	$z \leftarrow x \leftarrow y$	$z \leftarrow x \quad y$
volume			

Order polytope [Stanley'86]

partial order	$x \quad y \quad z$	$z \leftarrow x \leftarrow y$	$z \leftarrow x \quad y$
volume			
	$3! \int_0^1 \int_0^1 \int_0^1 \mathrm{d}z\ \mathrm{d}x\ \mathrm{d}y = 6$	$3! \int_0^1 \int_0^y \int_0^x \mathrm{d}z\ \mathrm{d}x\ \mathrm{d}y = 1$	$3! \int_0^1 \int_0^1 \int_0^x \mathrm{d}z\ \mathrm{d}x\ \mathrm{d}y = 3$

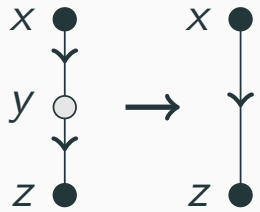
BITS decomposition

(B)ottom



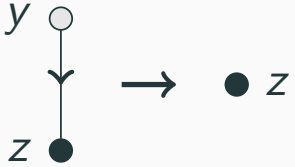
$\Psi' = \int_x^1 \Psi.dy$

(I)ntermediate



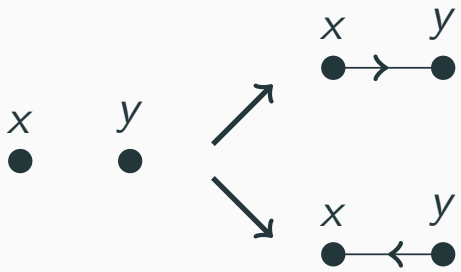
$\Psi' = \int_x^z \Psi.dy$

(T)op



$\Psi' = \int_0^z \Psi.dy$

(S)plit



$\Psi' = \Psi_{x \prec y} + \Psi_{y \prec x}$

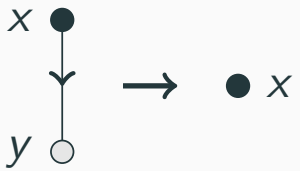
Theorem

BITS decomposition computes the volume of **any** poset.

The number of symbolic multivariate integrations may be **exponential** in the size of the poset.

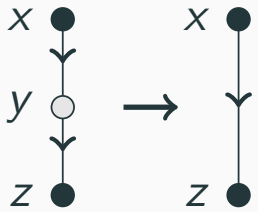
BITS decomposition

(B)ottom



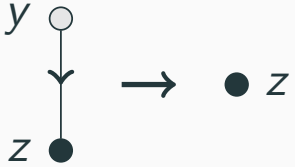
$\Psi' = \int_x^1 \Psi.dy$

(I)ntermediate



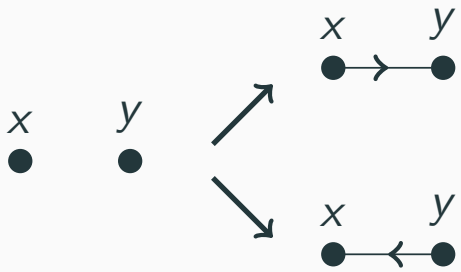
$\Psi' = \int_x^z \Psi.dy$

(T)op



$\Psi' = \int_0^z \Psi.dy$

(S)plit



$\Psi' = \Psi_{x \prec y} + \Psi_{y \prec x}$

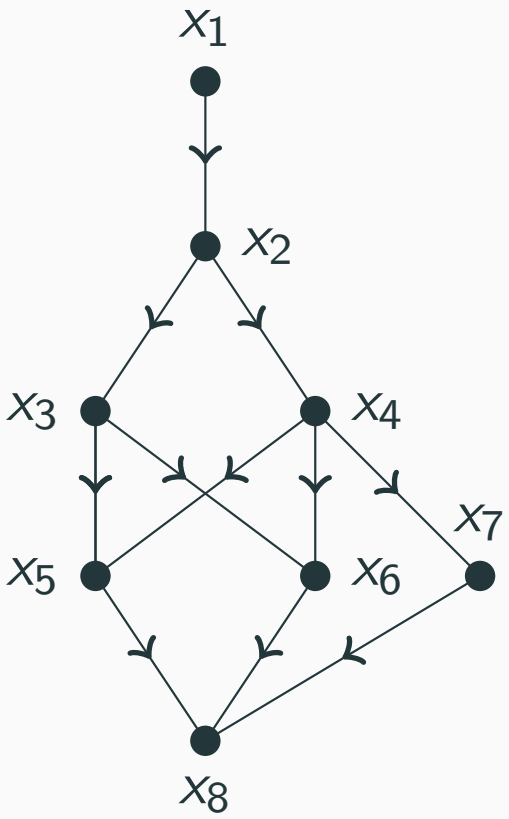
Theorem

BITS decomposition computes the volume of **any** poset.
The number of symbolic multivariate integrations may be **exponential** in the size of the poset.

Lemma

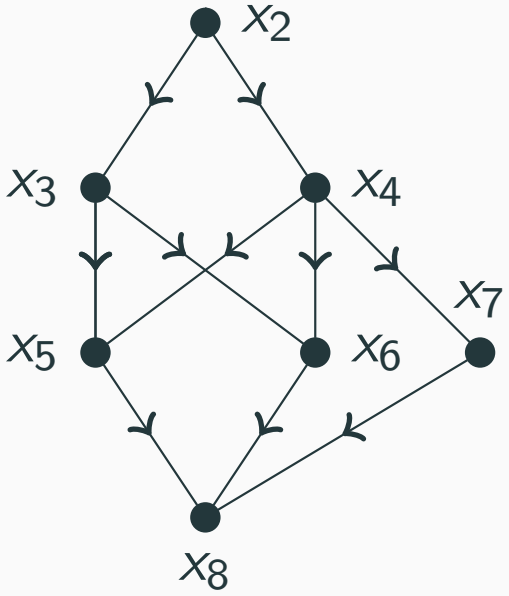
BIT rules are sufficient to decomposes any **cycle-free** poset.
The number of symbolic multivariate integrations is **linear** in the size of the poset.

Decomposition of the example



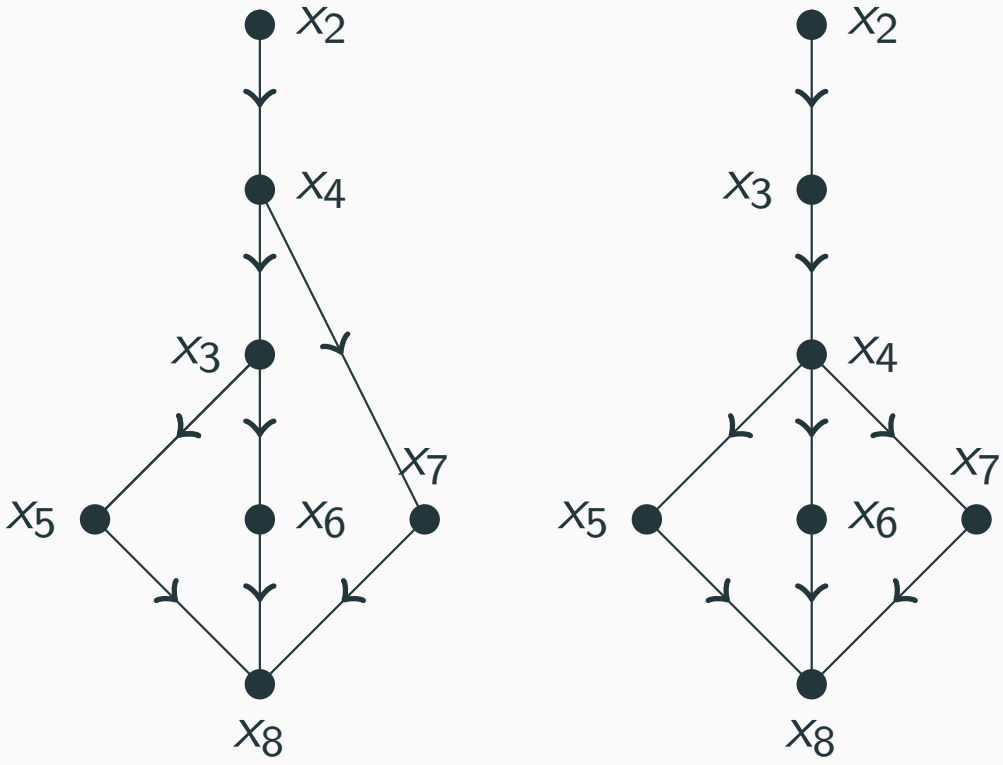
$$\Psi = 1$$

Decomposition of the example



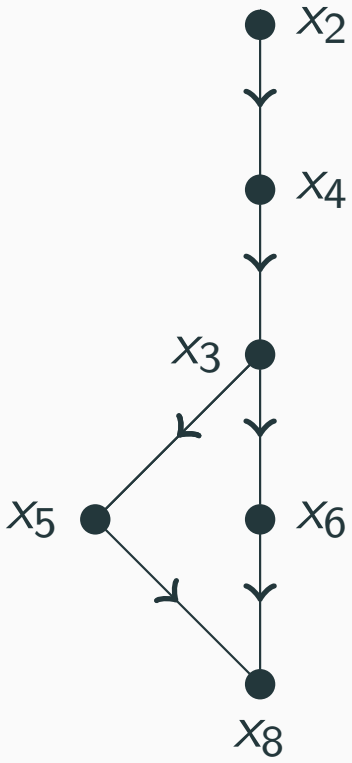
$$\psi' = \int_0^{x_2} \psi \mathrm{d}x_1$$

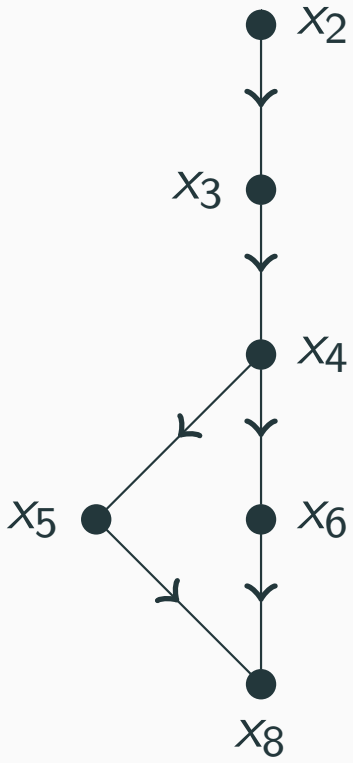
Decomposition of the example



$$\Psi'' = \Psi'_{x_4 \prec x_3} + \Psi'_{x_3 \prec x_4}$$

Decomposition of the example





$$\Psi''' = \int_{x_4}^{x_8} \Psi''_{x_4 \prec x_3} \mathrm{d}x_7 + \int_{x_4}^{x_8} \Psi''_{x_3 \prec x_4} \mathrm{d}x_7$$

Decomposition of the example

$$
\begin{aligned}
\tilde{\Psi} &= \int_0^{x_2} \Psi \mathrm{d}x_1 \\
&= \int_0^{x_2} \left(\Psi'_{x_4 \prec x_3} + \Psi'_{x_3 \prec x_4} \right) \mathrm{d}x_1 \\
&= \int_{x_4}^{x_8} \int_0^{x_2} \Psi''_{x_4 \prec x_3} \mathrm{d}x_1 \mathrm{d}x_7 + \int_{x_4}^{x_8} \int_0^{x_2} \Psi''_{\{x_3 \prec x_4\}} \mathrm{d}x_1 \mathrm{d}x_7 \\
&= \int_{x_3}^{x_8} \int_{x_4}^{x_8} \int_0^{x_2} \Psi'''_{x_4 \prec x_3} \mathrm{d}x_1 \mathrm{d}x_7 \mathrm{d}x_5 + \int_{x_4}^{x_8} \int_{x_4}^{x_8} \int_0^{x_2} \Psi'''_{x_3 \prec x_4} \mathrm{d}x_1 \mathrm{d}x_7 \mathrm{d}x_5 \\
&= \int_0^1 \int_{x_2}^1 \int_{x_4}^1 \int_{x_3}^1 \int_{x_6}^1 \int_{x_4}^{x_8} \int_{x_4}^{x_8} \int_0^{x_2} \mathrm{d}x_1 \mathrm{d}x_7 \mathrm{d}x_5 \mathrm{d}x_8 \mathrm{d}x_6 \mathrm{d}x_3 \mathrm{d}x_4 \mathrm{d}x_2 \\
&\quad + \int_0^1 \int_{x_2}^1 \int_{x_3}^1 \int_{x_4}^1 \int_{x_6}^1 \int_{x_4}^{x_8} \int_{x_4}^{x_8} \int_0^{x_2} \mathrm{d}x_1 \mathrm{d}x_7 \mathrm{d}x_5 \mathrm{d}x_8 \mathrm{d}x_6 \mathrm{d}x_4 \mathrm{d}x_3 \mathrm{d}x_2 \\
&= \frac{8+6}{8!} = \frac{14}{8!}.
\end{aligned}
$$

Random generation

Discrete to continuous

uniform random linear extension $\simeq$ uniform random simplex

$\simeq$ uniform random point of the polytope

Random generation

Key idea

If f is a continuous function from $[a, b]$ such that

$$\int_a^b f(x)\,\mathrm{d}x = C,$$

then $x \mapsto \frac{f(x)}{C}$ is the density of a random variable over $[a, b]$.

Random generation

Key idea

If f is a continuous function from $[a, b]$ such that

$$\int_a^b f(x)\,\mathrm{d}x = C,$$

then $x \mapsto \frac{f(x)}{C}$ is the density of a random variable over $[a, b]$.

Inversion method

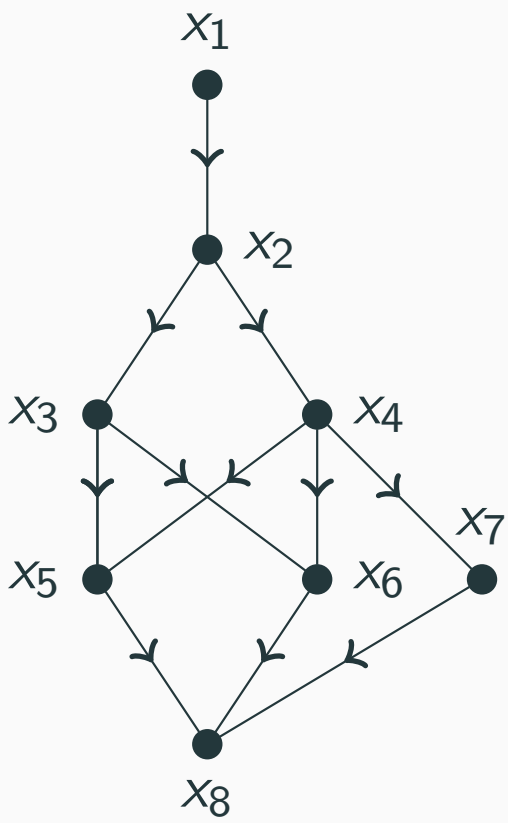
Let U be a r.v. of uniform law over $[a, b]$, then the solution X of

$$\int_a^X f(x)\,\mathrm{d}x = U,$$

is a r.v. of density f.

Uniformly sampling an execution in the example

$$\Psi = \Psi_{x_4 \prec x_3} + \Psi_{x_3 \prec x_4} = \frac{8+6}{8!}$$



Uniformly sampling an execution in the example

Volume:

$$\Psi_{x_4 \prec x_3} = \int_0^1 \int_{x_2}^1 \int_{x_4}^1 \int_{x_3}^1 \int_{x_6}^1 \int_{x_4}^{x_8} \int_{x_4}^{x_8} \int_0^{x_2} \mathrm{d}x_1 \mathrm{d}x_7 \mathrm{d}x_5 \mathrm{d}x_8 \mathrm{d}x_6 \mathrm{d}x_3 \mathrm{d}x_4 \mathrm{d}x_2$$

$$= \int_0^1 \frac{1}{453600} (x_2^6 - 6\,x_2^5 + 15\,x_2^4 - 20\,x_2^3 + 15\,x_2^2 - 6\,x_2 + 1) x_2 \; \mathrm{d}x_2$$

Equation of inversion

$$\int_0^{X_2} \frac{1}{90} (x_2^6 - 6\,x_2^5 + 15\,x_2^4 - 20\,x_2^3 + 15\,x_2^2 - 6\,x_2 + 1) x_2 \; \mathrm{d}x_2 = 0.13$$

Sampled point

$$X_2 = 0.08$$

Uniformly sampling an execution in the example

Equation of inversion

$$\int_{X_2}^{X_4} -\frac{1}{15}\,(x_4^5 - 5\,x_4^4 + 10\,x_4^3 - 10\,x_4^2 + 5\,x_4 - 1)X_2\;\mathrm{d}x_4 = 0.84$$

Sampled point

$$X_2 = 0.08$$

$$X_4 = 0.32$$

Uniformly sampling an execution in the example

Equation of inversion

$$\int_{X_4}^{X_3} -\frac{1}{12}\left(x_3^4 - 6\,x_3^2 - 2\left(2\,x_3^3 - 6\,x_3^2 + 3\,x_3\right)X_4 - 2\left(3\,x_3 - 2\right)X_4 + 8\,x_3 - 3\right)X_2 \,\mathrm{d}x_3 = 0.76$$

Sampled point

$$X_2 = 0.08$$

$$X_4 = 0.32$$

$$X_3 = 0.54$$

Uniformly sampling an execution in the example

Equation of inversion

$$\int_{X_3}^{x_6} -\frac{1}{6}\left(6\,X_3X_4x_6 - 3\,(X_3+X_4)x_6^2 + 2\,x_6^3 - 3\,(2\,X_3-1)X_4 + 3\,X_3 - 2)x_2\right.\,\mathrm{d}x_6 = 0.25$$

Sampled point

$$X_2 = 0.08$$
$$X_4 = 0.32$$
$$X_3 = 0.54$$
$$X_6 = 0.62$$

Uniformly sampling an execution in the example

Sampled point

$$X_2 = 0.08$$
$$X_4 = 0.32$$
$$X_3 = 0.54$$
$$X_6 = 0.62$$
$$X_8 = 0.89$$
$$X_5 = 0.76$$
$$X_7 = 0.58$$
$$X_1 = 0.06$$

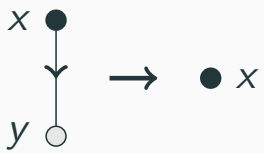
Linear extension

$$x_1 \prec x_2 \prec x_4 \prec x_3 \prec x_7 \prec x_6 \prec x_5 \prec x_8$$

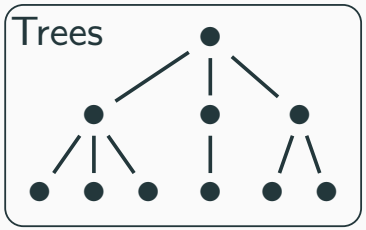
Complexity results

Strategies for the decomposition

B(ottom)

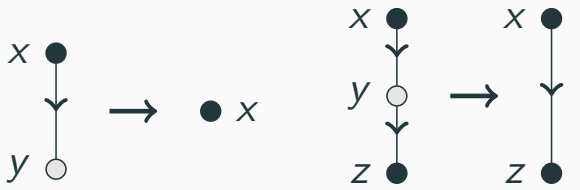


$$\Psi' = \int_x^1 \Psi.dy$$

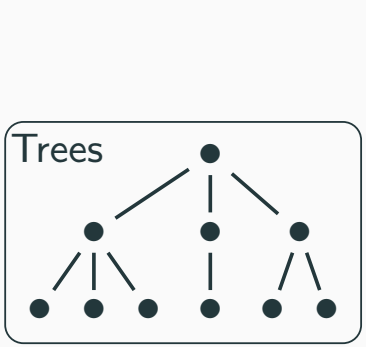


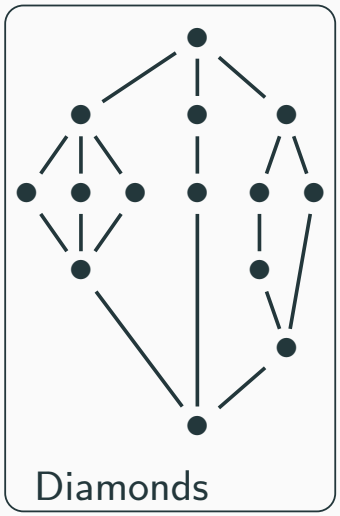
Strategies for the decomposition

B(ottom) I(ntermediate)



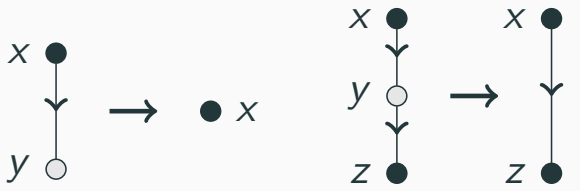
$$\Psi' = \int_x^1 \Psi.dy \qquad \Psi' = \int_x^z \Psi.dy$$



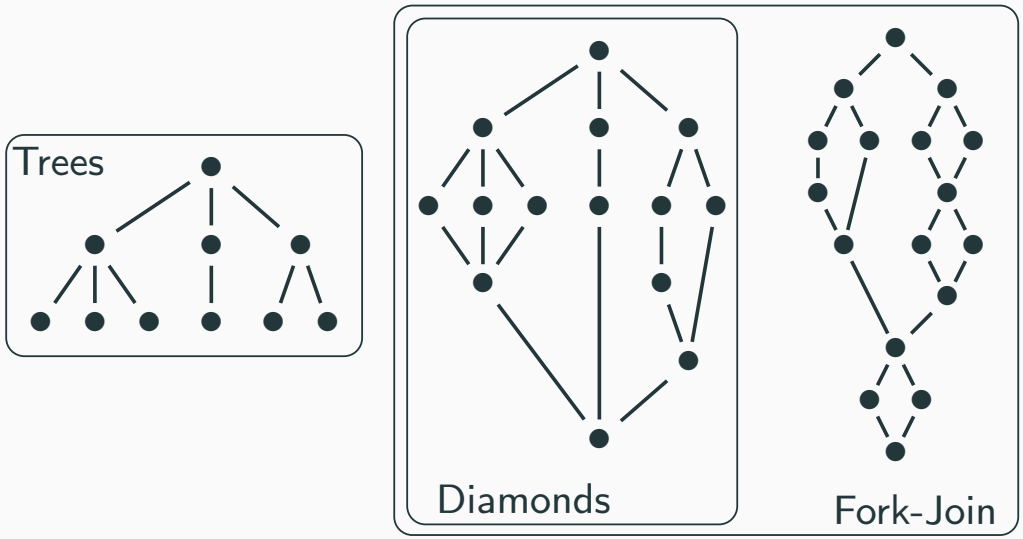


Strategies for the decomposition

B(ottom) I(ntermediate)

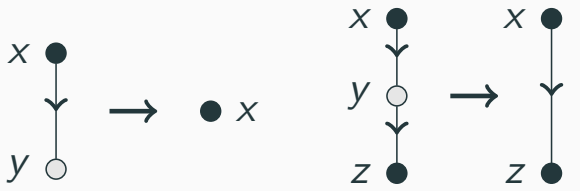


$$\Psi' = \int_x^1 \Psi.dy \qquad \Psi' = \int_x^z \Psi.dy$$

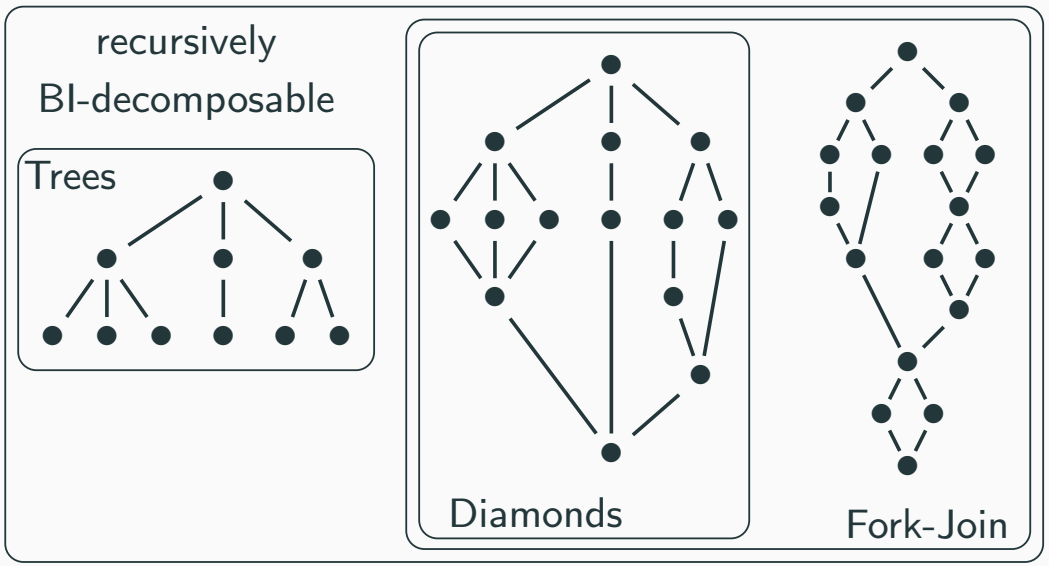


Strategies for the decomposition

B(ottom) I(ntermediate)

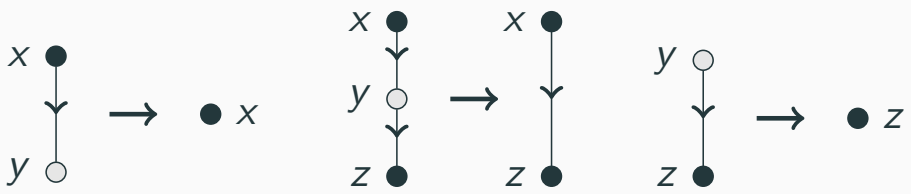


$$\Psi' = \int_x^1 \Psi.dy \qquad \Psi' = \int_x^z \Psi.dy$$

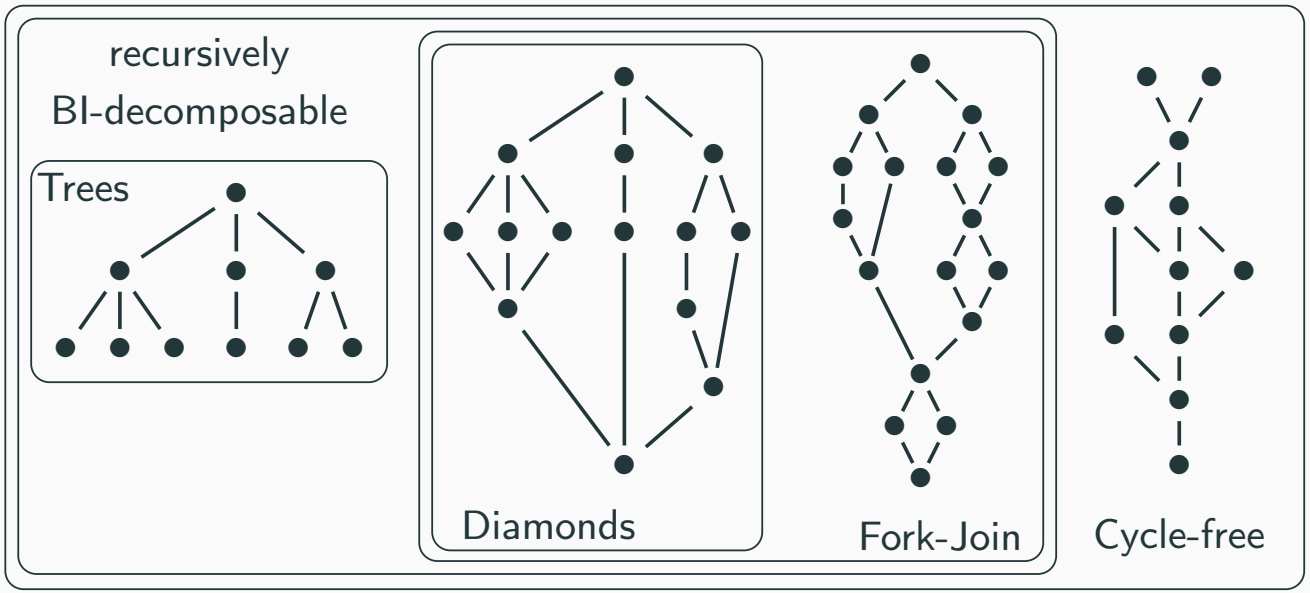


Strategies for the decomposition

B(ottom) I(ntermediate) T(op)

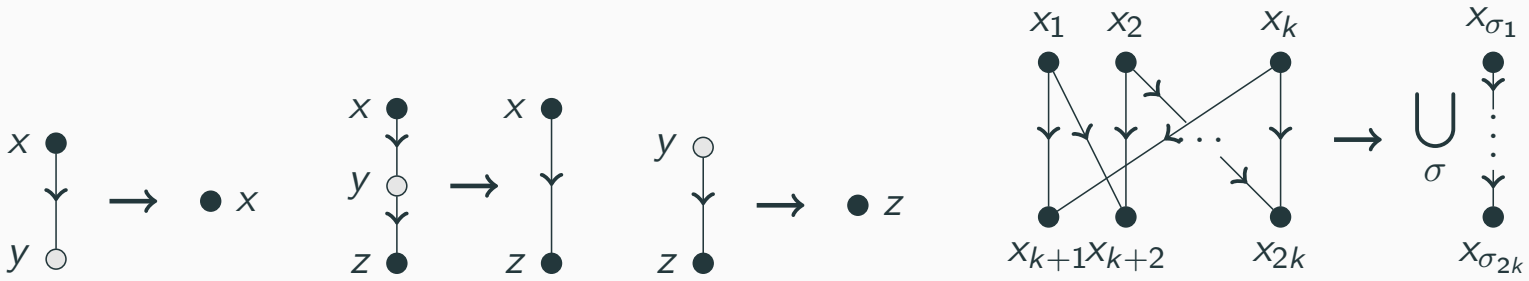


$$\Psi' = \int_x^1 \Psi.dy \qquad \Psi' = \int_x^z \Psi.dy \qquad \Psi' = \int_0^z \Psi.dy$$

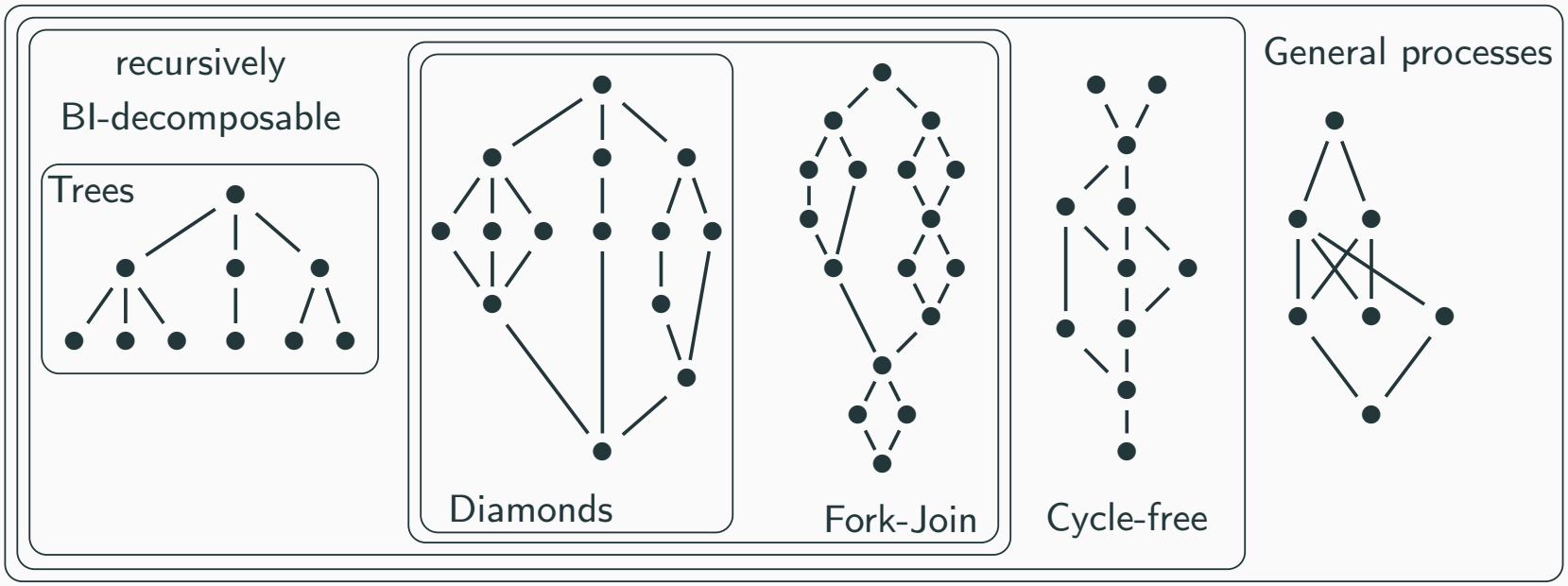


Strategies for the decomposition

B(ottom) I(ntermediate) T(op) C(ycle)



$$\Psi' = \int_x^1 \Psi.dy \qquad \Psi' = \int_x^z \Psi.dy \qquad \Psi' = \int_0^z \Psi.dy \qquad \Psi' = \sum_{\sigma \text{ a total order}} \Psi_\sigma$$



(Micro-) Benchmark

FJ size	$\sharp\mathcal{LE}$	FJ **gen**	**(count)**	bit **gen**	**(count)**	cftp **gen**
10	19	1.10^{-5} s	$(2.10^{-4}$ s)	6.10^{-4} s	(0.03 s)	0.04 s
30	10^9	2.10^{-5} s	$0(2.10^{-4}$ s)	0.02 s	(0.03 s)	1.8 s
40	$6 \cdot 10^6$	4.10^{-5} s	$(3.10^{-4}$ s)	3.5 s	(5.2 s)	5.6 s
63	$4 \cdot 10^{29}$	5.10^{-4} s	(0.03 s)	Mem. crash	(Crash)	55 s
217028	$2 \cdot 10^{292431}$	8.11 s	(3.34 s)	Mem. crash	(Crash)	Timeout

Arch size	$\sharp\mathcal{LE}$	Arch **gen**	**(count)**	bit **gen**	**(count)**	cftp **gen**
10:2	43	2.10^{-5} s	$(4.10^{-5}$ s)	0.002 s	6.10^{-6} s)	0.04 s
30:2	$9.8 \cdot 10^8$	0.003 s	(0.0009 s)	7.10^{-6} s	(0.0004 s)	1.5 s
30:4	$6.9 \cdot 10^{10}$	0.001 s	(0.005 s)	7.10^{-5} s	(0.004 s)	2.5 s
100:2	$1.3 \cdot 10^{32}$	0.75 s	(0.16 s)	Mem. crash	(Crash)	5.6 s *
100:32	$1 \cdot 10^{53}$	2.7 s	(0.17 s)	Mem. crash	(Crash)	5.9 s *
200:66	10^{130}	54 s	(31 s)	Mem. crash	(Crash)	Timeout

$\Rightarrow$ All the code available at

`https://gitlab.com/ParComb/combinatorics-barrier-synchro`

Parallelism and Synchronization

Uniform random generation of executions | Counting

Diamonds [BoDiFoGeHw16]
$O(n \log n)$ | $O(n)$

↓

Fork-Join processes [BoDiGePe17]
$O(n\sqrt{n})$ | $O(n)$

($\|$)-Trees [BoGePe16]
$O(n \log n)$ | $O(n)$

Series Parallel processes
$O(n\sqrt{n})$ [BoDiGePe17] | $O(n)$ [Mö87]

↓

Cycle-free processes
$O(n^3 \log n)$ [Hu06] | $O(?)$: Conjecture by [Ha89]

↓

All processes
$O(n^3 \log n)$ [Hu06] | $\sharp$P-complet [BrWi91]

n = process size