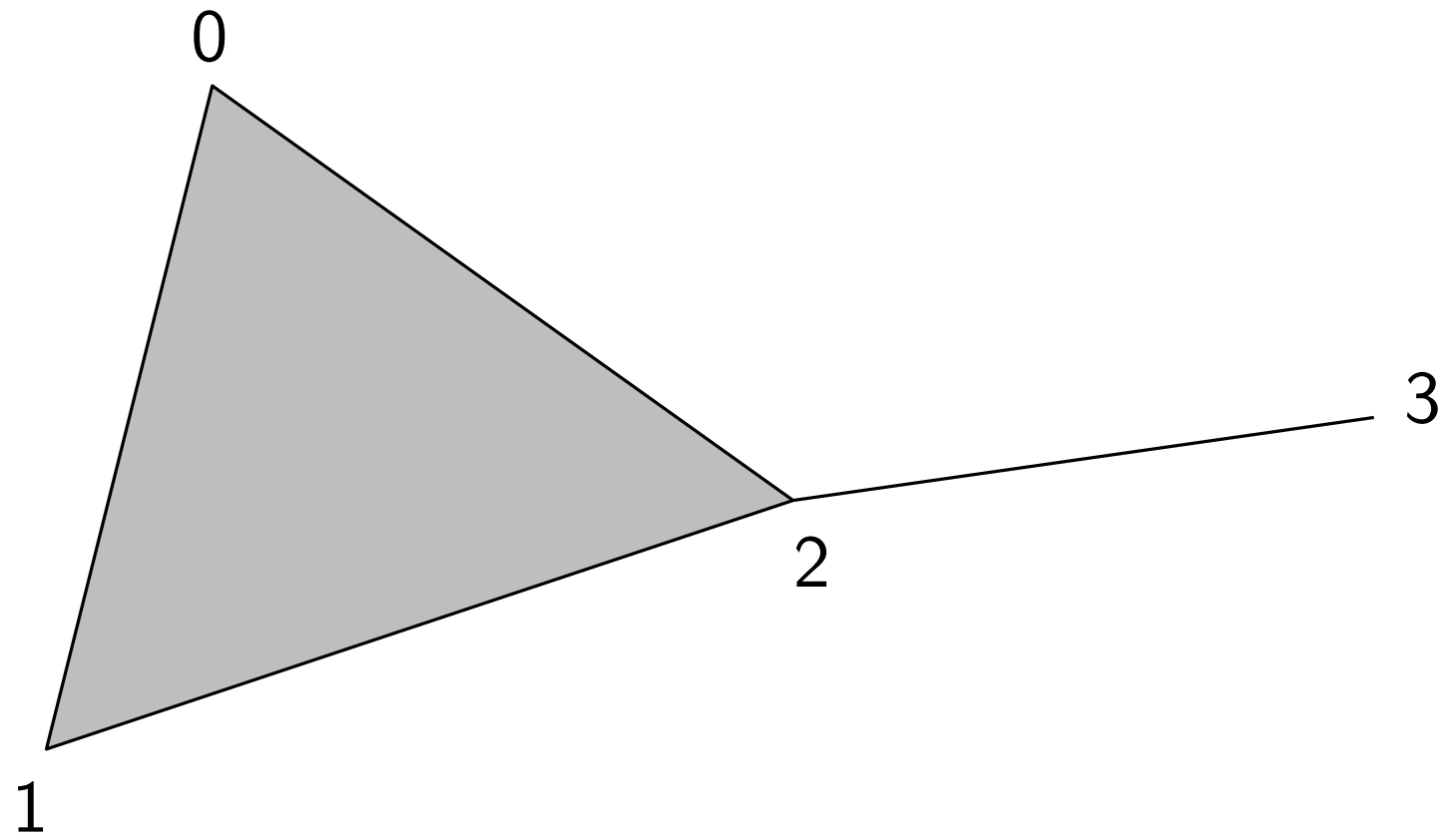
Simplex Array List

Marc Glisse

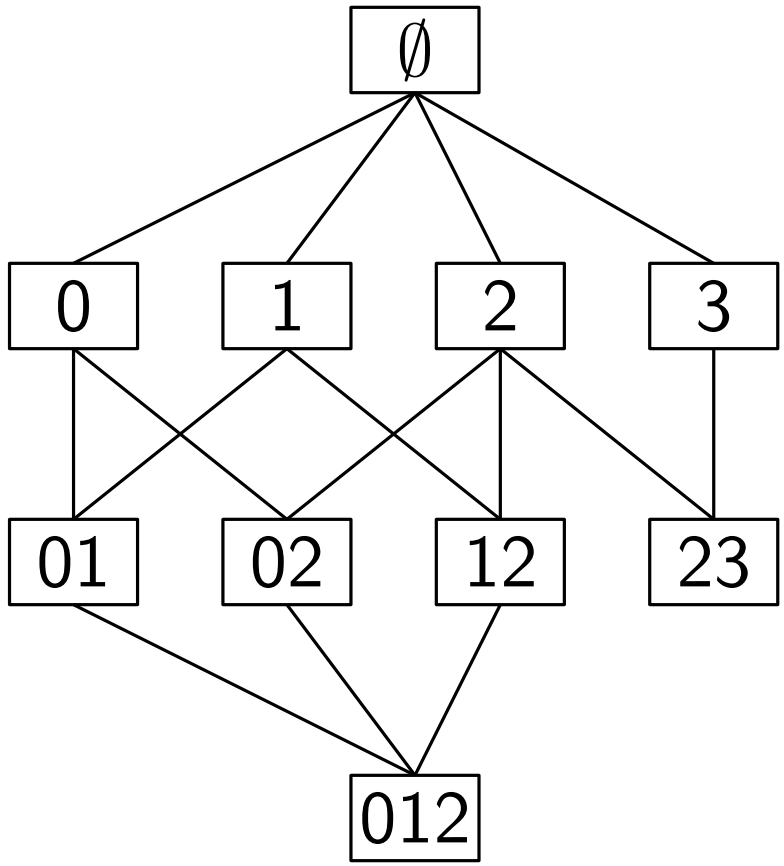
Simplicial complex

List of simplices: 0, 1, 2, 3, 01, 02, 12, 23, 012.



Hasse diagram

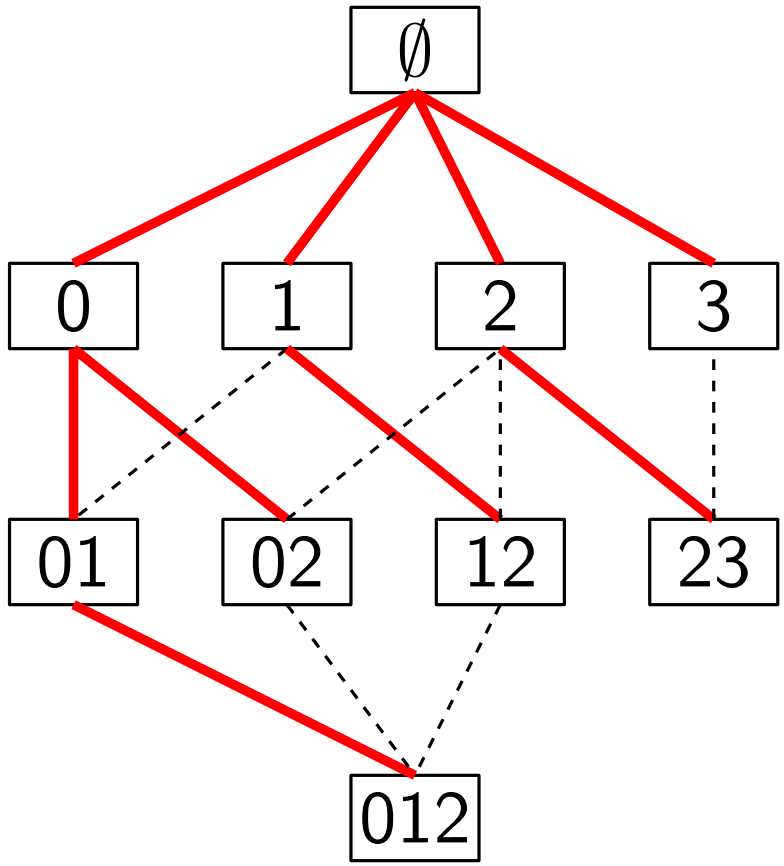
List of simplices: 0, 1, 2, 3, 01, 02, 12, 23, 012.



Very large, many arrows.

Simplex tree (1)

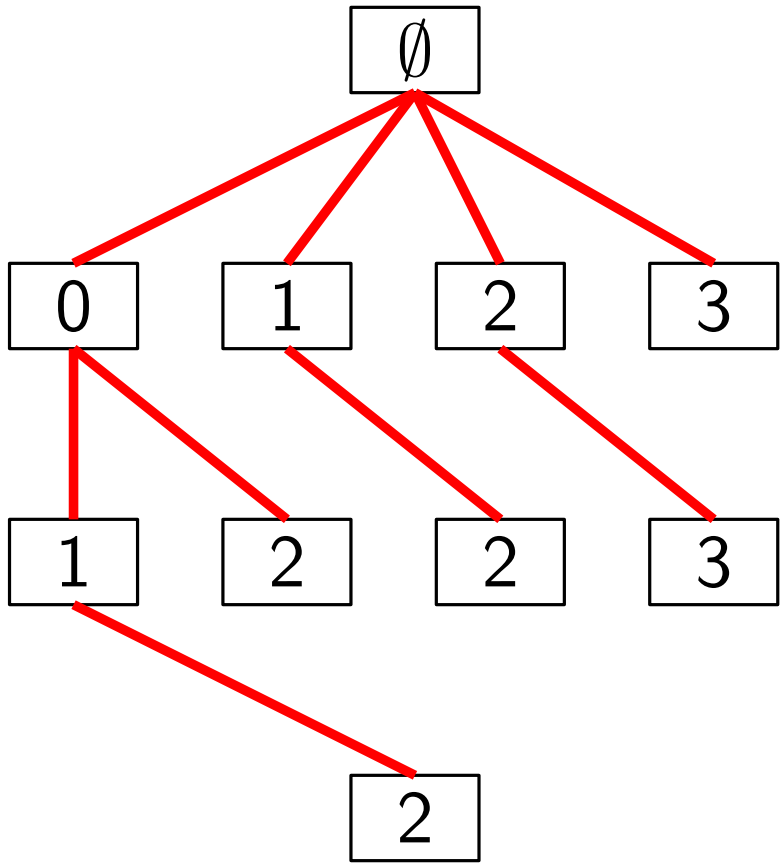
List of simplices: 0, 1, 2, 3, 01, 02, 12, 23, 012.



Keep only a spanning (prefix) tree.

Simplex tree (2)

List of simplices: 0, 1, 2, 3, 01, 02, 12, 23, 012.



Remove label redundancy (trie).

Simplex tree (3)

Size: O(number of simplices).

Optimal size if we want to store data per simplex (filtration value).

Some operations can still be done in reasonable time (find, boundary, insert...), though memory accesses are expensive.

Extra heuristic to speed up other operations (coboundary, removal, edge contraction): link nodes sharing the same label.

Automaton minimization: *find* still works.

Maximal simplices

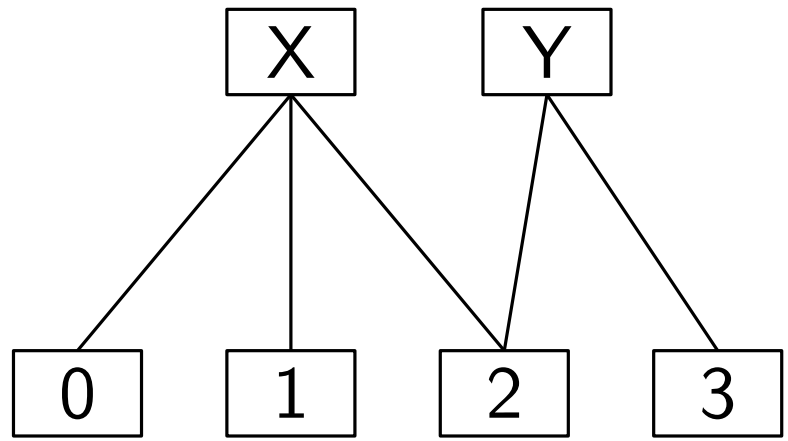
List of simplices: 0, 1, 2, 3, 01, 02, 12, 23, 012.

List of maximal simplices: 012, 23.

Much more compact.

How do we use it?

SAL-0 definition

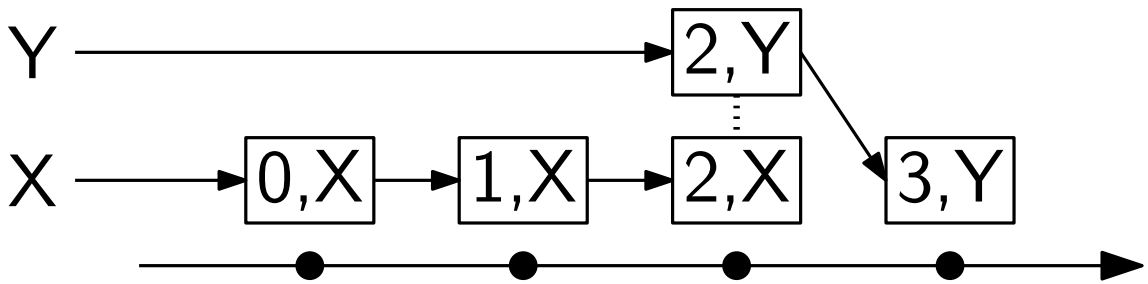


X is a label for simplex 012 and Y for 23.

Make the arrows bidirectional!

Key assumption: each vertex belongs to a small number of maximal simplices (max. Γ_0).

Practical representation: array of lists (of maximal simplices).



SAL-0 operations

find(σ): Compute the intersection of the lists of maximal simplices of the vertices of σ.

> Edge 12 is in the complex since 1 and 2 belong to the same maximal simplex X.

insert(σ): Find. Insert. Remove simplices that stop being maximal.

> Look for such a simplex in the lists of maximal simplices of the vertices of σ.

remove(σ): Find. Remove τ. For each vertex v of σ, insert $\tau \setminus \{v\}$.

edge contraction, etc.

Complexity

All operations are local.

No complexity involves the total number of maximal simplices.

Complexity

All operations are local.

No complexity involves the total number of maximal simplices.

find(1234): Intersection of 4 lists.

Speed-up: precompute pairwise intersections.

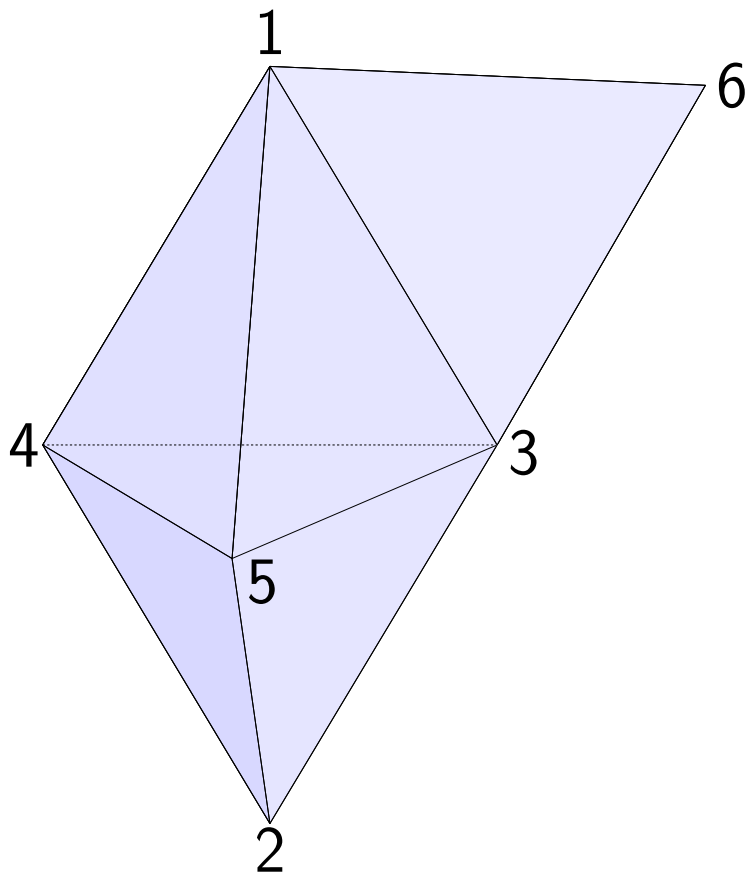
Intersect list(12) with list(34).

Fewer lists.

Shorter lists: $\Gamma_1 < \Gamma_0$.

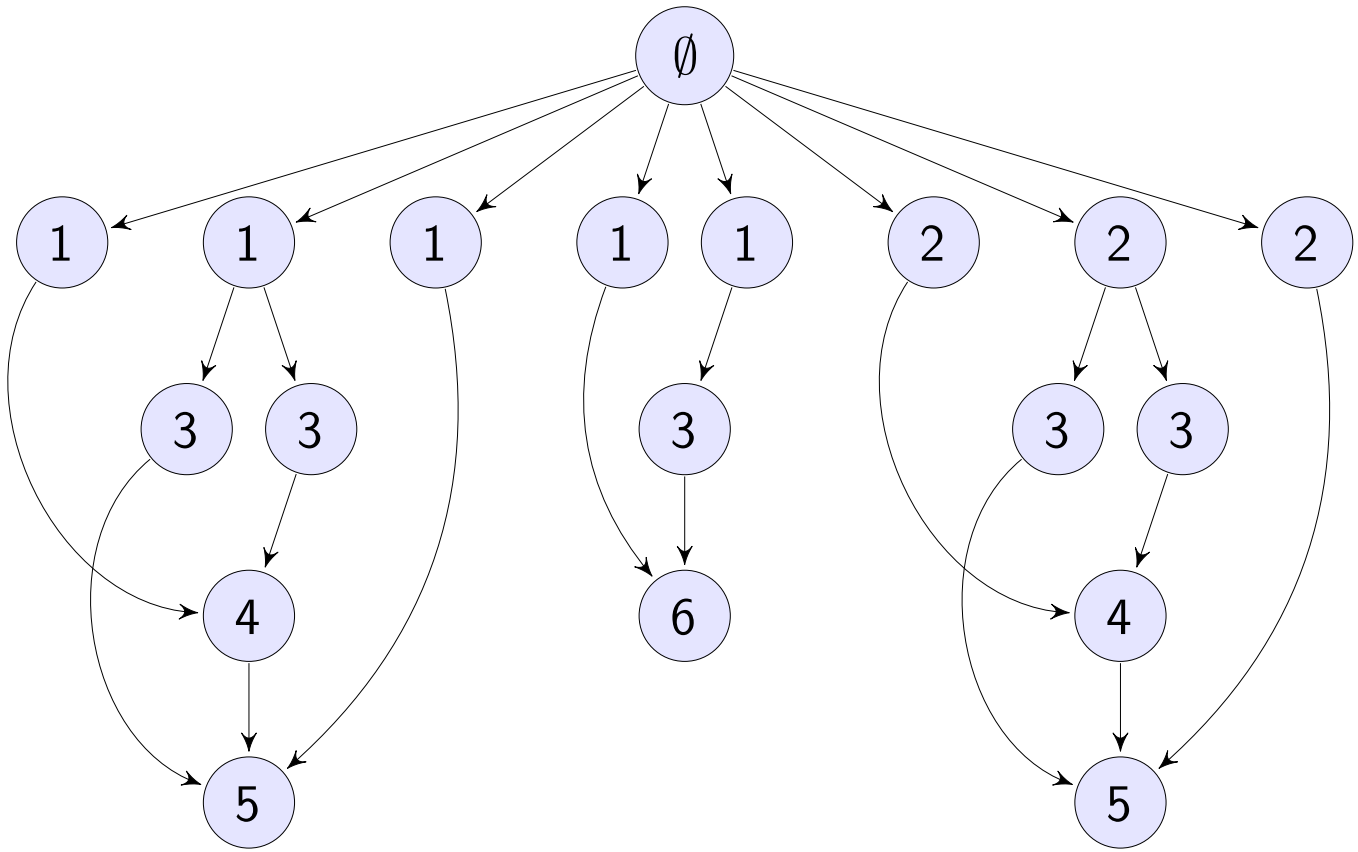
SAL-1

New example.

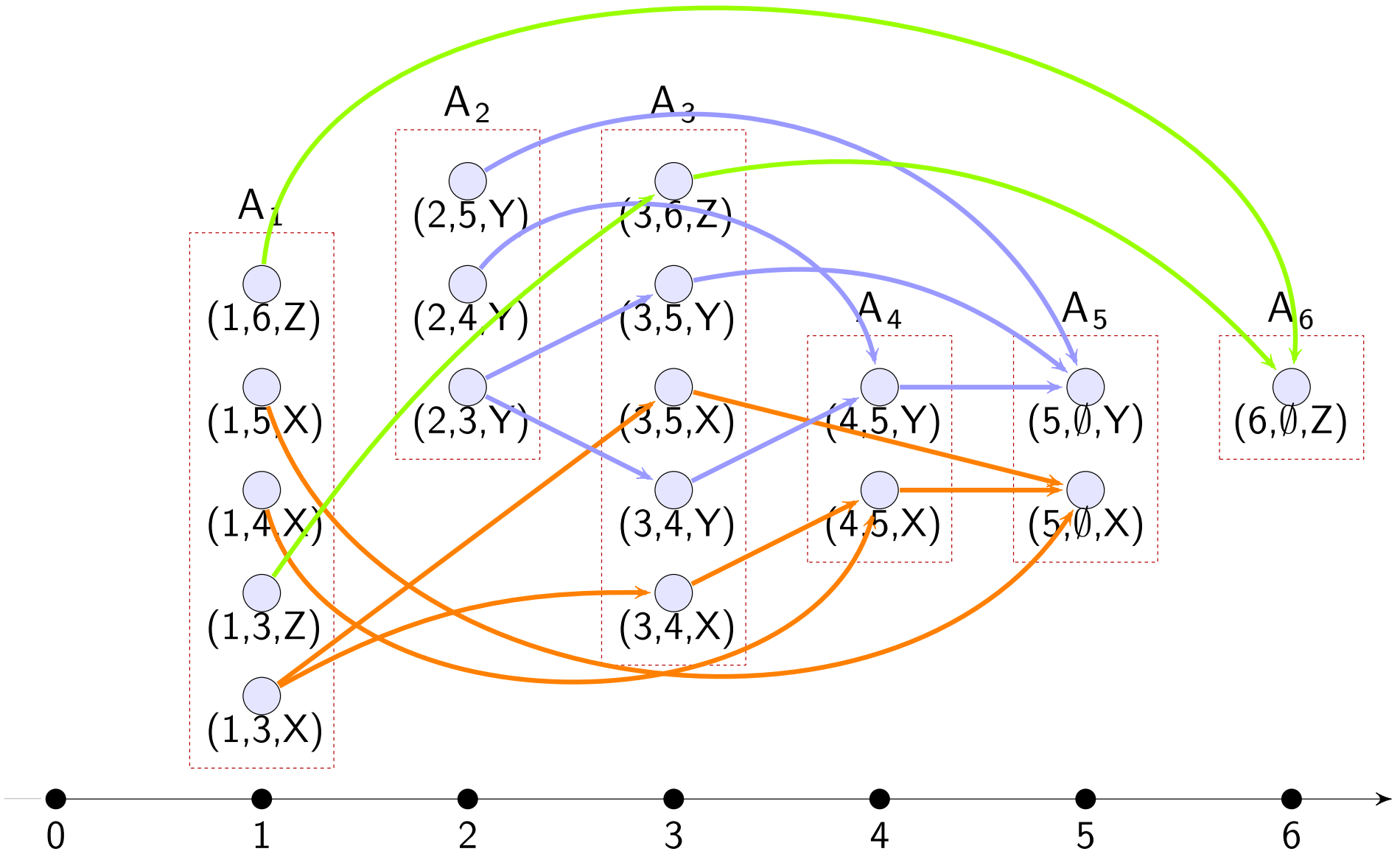


SAL-1

One (inconvenient) representation.



SAL-1


A1
A2
A3
A4
A5
A6
(1,6,Z)
(1,5,X)
(1,4,X)
(1,3,Z)
(1,3,X)
(2,5,Y)
(2,4,Y)
(2,3,Y)
(3,6,Z)
(3,5,Y)
(3,5,X)
(3,4,Y)
(3,4,X)
(4,5,Y)
(4,5,X)
(5,∅,Y)
(5,∅,X)
(6,∅,Z)
0
1
2
3
4
5
6

SAL-1: arrows

The size is dominated by the arrows: $\Theta(d^3)$ for a simplex.

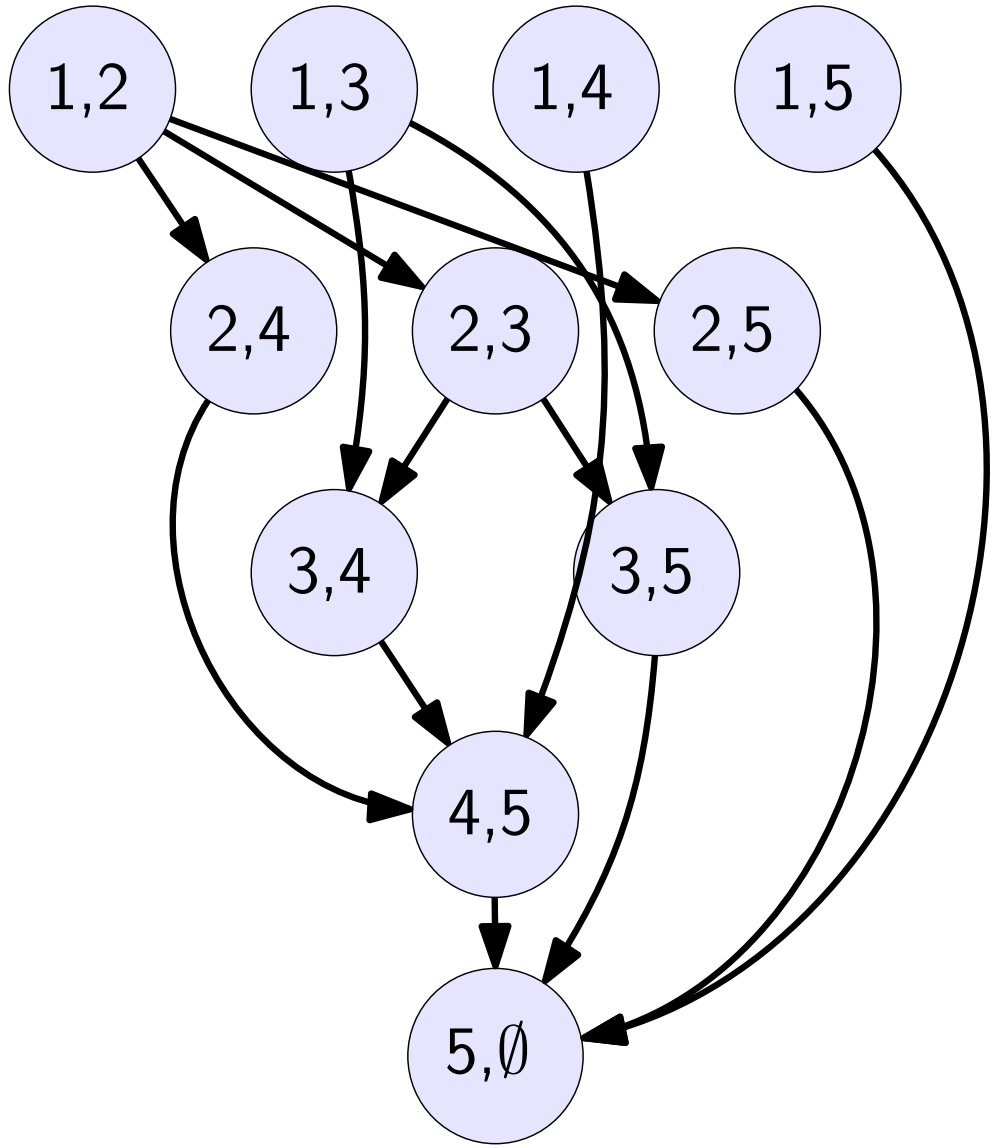
What are arrows used for?

Get the list of vertices of a maximal simplex.

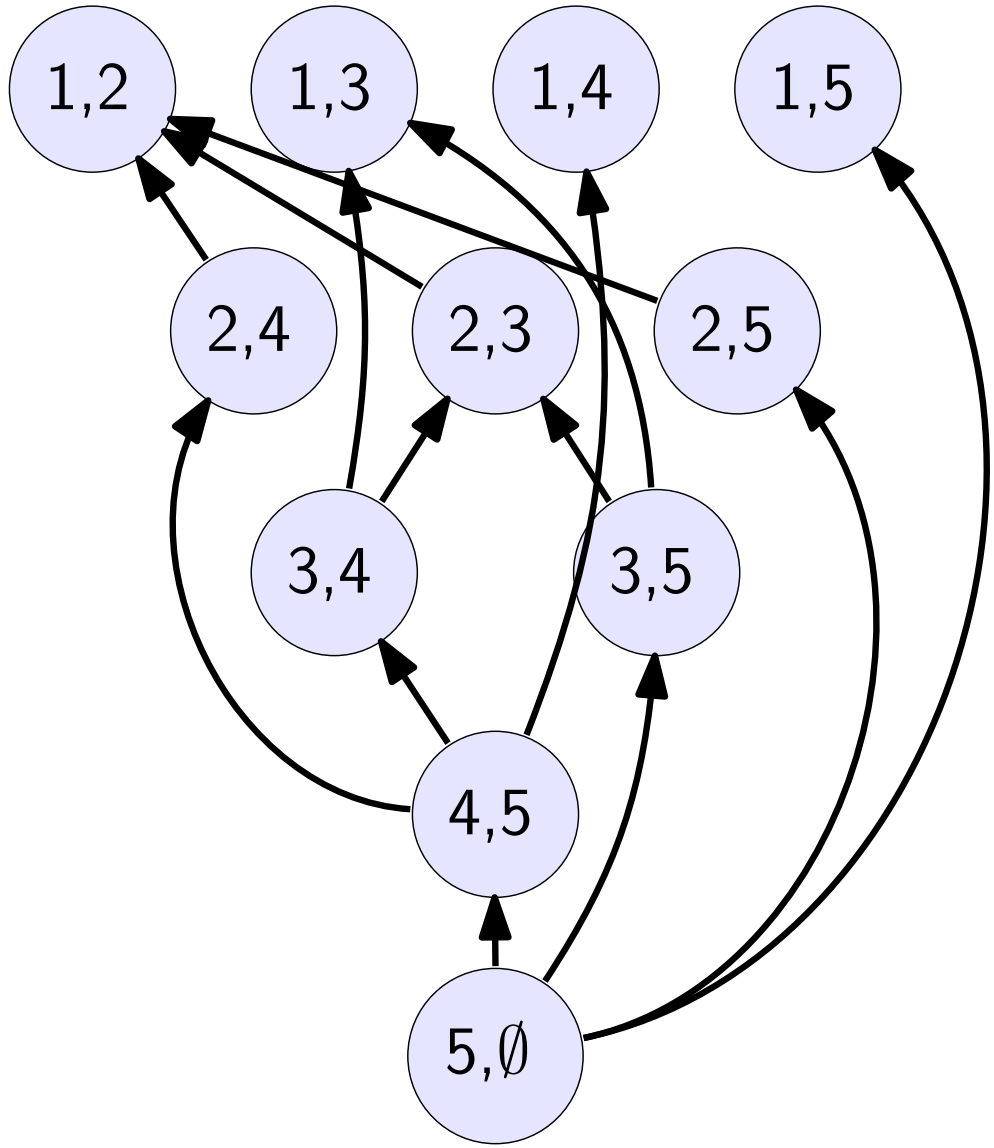
Get the list of nodes of a maximal simplex (for removal).

Use better arrows...

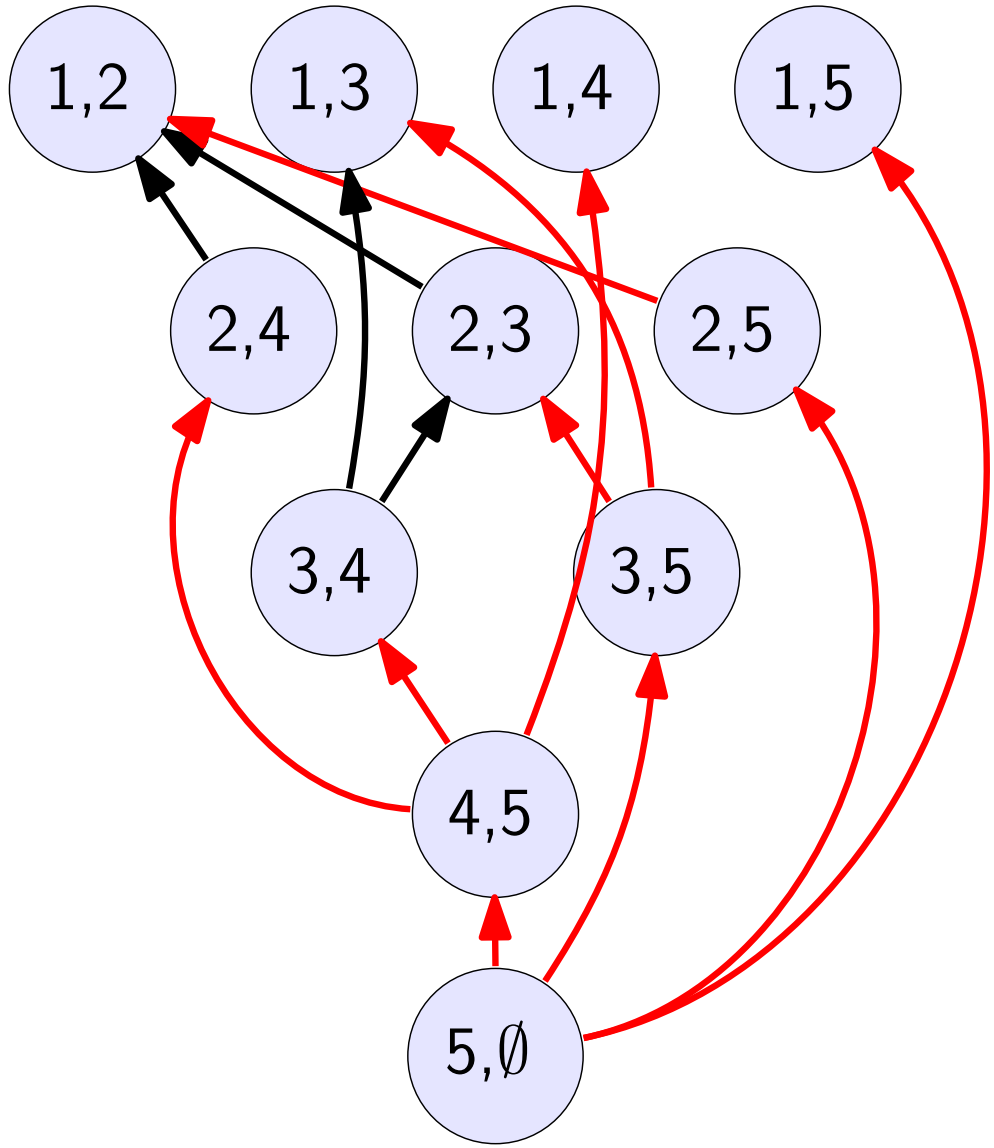
Better arrows for SAL-1



Better arrows for SAL-1



Better arrows for SAL-1



Experiments

Build a Rips complex, insert/remove random simplices, contract random edge.

SAL-0 smaller than SAL-1 smaller than simplex tree.

Simplex tree easily runs out of memory.

Fewer arrows in SAL-1: small difference (factor 2 in dim 30).

SAL-0 faster in high dimension, SAL-1 in low dimension.

Rips faster with simplex tree when it fits in memory.

More

Comparison with Skeleton-blockers.

Graph + minimal simplices not in the complex.