Space and Congruence Compression of Proofs

Andreas Fellner

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

European Master in Computational Logic

Master Thesis Defense
Vienna, 23rd of September 2014

Space and Congruence Compression of Proofs

Space and Congruence Compression of **Proofs**

Resolution

Resolution Rule

$$\frac{C \lor x \quad D \lor \neg x}{C \lor D}$$

Resolution

Resolution Rule

$$\begin{tikzcd} C, x & & D, \neg x \\ & C, D \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$

Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$

$\neg x_1$

Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$

$$\begin{tikzcd}
x_1, x_2, x_3 & & \neg x_1 \\
& x_2, x_3 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

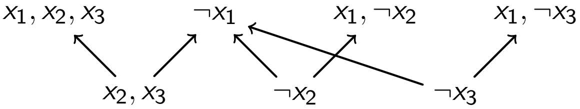
Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$

$$\begin{tikzcd}
x_1, x_2, x_3 & & \neg x_1 & & x_1, \neg x_2 \\
& x_2, x_3 \arrow[ul] \arrow[ur] & & \neg x_2 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

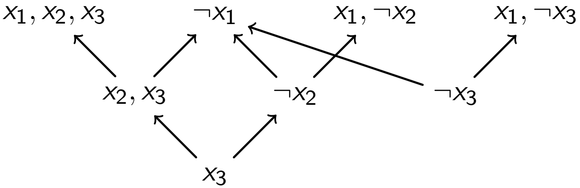
Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$



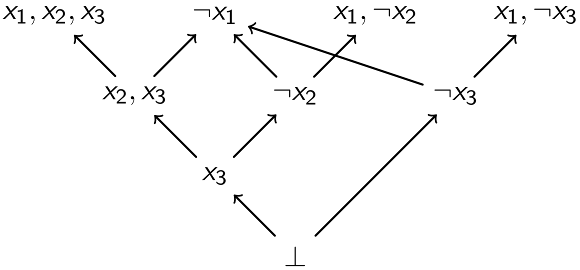
Example

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1)$$



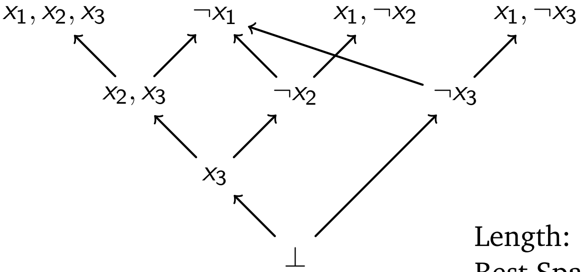
Example

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1)$$



Example

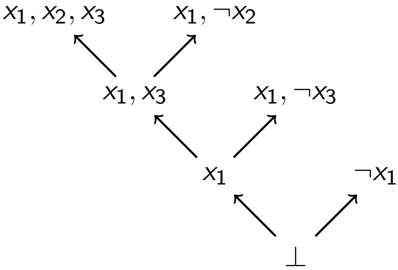
$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1)$$



Length: 9
Best Space: 4

Example

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1)$$



Length: 7
Best Space: 3

Space and Congruence **Compression** of Proofs

Proof Compression

- Smaller unsat cores, interpolants
- Easier proof processing
- Smaller proofs libraries
- Easier trusted interaction of deductive systems
- Proof generalization
- Proof carrying code

Synthesizing Multiple Boolean Functions using Interpolation on a Single Proof

- Georg Hofferek, et al, 2013, TU Graz

Synthesizing Multiple Boolean Functions using Interpolation on a Single Proof

- Georg Hofferek, et al, 2013, TU Graz

Method

1. Formulate problem in SMT theory of uninterpreted functions
2. Obtain proof of unsatisfiability from SMT solver
3. Transform proof (local first, colorable)
4. Extract a single n-interpolant from the proof
5. Extract multiple interpolants the single interpolant

Synthesizing Multiple Boolean Functions using Interpolation on a Single Proof

- Georg Hofferek, et al, 2013, TU Graz

Method

1. Formulate problem in SMT theory of uninterpreted functions
2. Obtain proof of unsatisfiability from SMT solver
3. Transform proof (local first, colorable)
4. Extract a single *n*-interpolant from the proof
5. Extract multiple interpolants the single interpolant

- Worst case exponential runtime in proof size

Synthesizing Multiple Boolean Functions using Interpolation on a Single Proof

- Georg Hofferek, et al, 2013, TU Graz

Method

1. Formulate problem in SMT theory of uninterpreted functions
2. Obtain proof of unsatisfiability from SMT solver
3. Transform proof (local first, colorable)
4. Extract a single *n*-interpolant from the proof
5. Extract multiple interpolants the single interpolant

- Worst case exponential runtime in proof size

Proof Compression using Skeptik

- Input proof: 1,870,407 nodes
- Output proof: 868,760 nodes (53,6% compression)

Space and **Congruence** Compression of Proofs

Example

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Example

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Unsatisfiable!

Example

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Unsatisfiable!

Proof

Equality is transitive, therefore from $f(a) = a$, $a = b$ and $b = f(b)$ follows $f(a) = f(b)$, which contradicts $f(a) \neq f(b)$

Example

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Unsatisfiable!

Proof

Equality is transitive, therefore from $f(a) = a$, $a = b$ and $b = f(b)$ follows $f(a) = f(b)$, which contradicts $f(a) \neq f(b)$

A different Proof

$f(.)$ is a function, therefore from $a = b$ follows $f(a) = f(b)$, which contradicts $f(a) \neq f(b)$

A proof

$$\begin{tikzcd}
f(a) \neq a, a \neq b, b \neq f(b), f(a) = f(b) & & f(a) = a & & \\
& a \neq b, b \neq f(b), f(a) = f(b) \arrow[ul] \arrow[ur] & & a = b & \\
& & b \neq f(b), f(a) = f(b) \arrow[ul] \arrow[ur] & & b = f(b) \\
& f(a) \neq f(b) & & f(a) = f(b) \arrow[ul] \arrow[ur] & \\
& & \bot \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

A different proof

$$\begin{tikzcd}
 & & a \neq b, f(a) = f(b) & & a = b \\
 & f(a) \neq f(b) & & f(a) = f(b) \arrow[ul] \arrow[ur] & \\
 & & \bot \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

Congruence Closure

Congruence Closure

Ground Terms

- Constants $a, b, c, \ldots$
- Compound Terms $f(t_1, \ldots, t_n)$

Congruence Closure

Ground Terms

- Constants $a, b, c, \ldots$
- Compound Terms $f(t_1, \ldots, t_n)$

Congruence Relation R

- Reflexive: $\forall t (t, t) \in R$
- Symmetric: $(s, t) \in R \Rightarrow (t, s) \in R$
- Transitive: $(t_1, t_2) \in R \ldots (t_{m-1}, t_m) \in R \Rightarrow (t_1, t_m) \in R$
- Compatible: $\forall i (t_i, s_i) \in R \Rightarrow (f(t_1, \ldots, t_n), f(s_1, \ldots, s_n)) \in R$

Congruence Closure

Ground Terms

- Constants $a, b, c, \ldots$
- Compound Terms $f(t_1, \ldots, t_n)$

Congruence Relation R

- Reflexive: $\forall t (t, t) \in R$
- Symmetric: $(s, t) \in R \Rightarrow (t, s) \in R$
- Transitive: $(t_1, t_2) \in R \ldots (t_{m-1}, t_m) \in R \Rightarrow (t_1, t_m) \in R$
- Compatible: $\forall i (t_i, s_i) \in R \Rightarrow (f(t_1, \ldots, t_n), f(s_1, \ldots, s_n)) \in R$

Congruence Closure E^* of set of equations E

- Smallest Congruence Relation containing E
- Computable in $O(n \log(n))$
- E is explanation for $(s, t) \in E^*$

Example revisited

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Example revisited

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Explanation for $f(a) = f(b)$

$\{ \; f(a) = a, \; a = b \; , b = f(b) \; \}$

Example revisited

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Explanation for $f(a) = f(b)$

$\{ \quad a = b \, , b = f(b) \}$

Example revisited

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Explanation for $f(a) = f(b)$

$\{ \quad a = b \quad \}$

Example revisited

Knowledge

1. $f(a) = a$
2. $a = b$
3. $b = f(b)$
4. $f(a) \neq f(b)$

Explanation for $f(a) = f(b)$

$\{ \quad a = b \quad \}$

Short explanation $\rightsquigarrow$ short (sub)proof

Short Explanation Decision Problem

Given a set of input equations E, a target equation $s = t$ and $k \in \mathbb{N}$, does there exist an explanation $E' \subseteq E$ of $s = t$ with $|E'| \leq k$?

Short Explanation Decision Problem

Given a set of input equations E, a target equation $s = t$ and $k \in \mathbb{N}$, does there exist an explanation $E' \subseteq E$ of $s = t$ with $|E'| \leq k$?

NP-complete

NP-completeness proof sketch

From a propositional logic formula Φ obtain ...

- a set of equations E_Φ
- a target equation $s_\Phi = t_\Phi$
- $k_\Phi \in \mathbb{N}$

such that ...

Φ is satisfiable if and only if there is an explanation $E' \subseteq E_\Phi$ of $s_\Phi = t_\Phi$ with $|E'| \leq k_\Phi$

NP-completeness proof sketch example

Formula

$$(x_1 \lor x_2 \lor \neg x_3) \land (\neg x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2)$$

Translation to equations

NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations

$\bot_1$ —— $\hat{x}_1$ —— $\top_1$ $\quad$ $\bot_2$ —— $\hat{x}_2$ —— $\top_2$ $\quad$ $\bot_3$ —— $\hat{x}_3$ —— $\top_3$

NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations

$$\bot_1 \;—\; \hat{x}_1 \;—\; \top_1 \qquad \bot_2 \;—\; \hat{x}_2 \;—\; \top_2 \qquad \bot_3 \;—\; \hat{x}_3 \;—\; \top_3$$

$$\begin{array}{lll} & & t_1(\hat{x}_1) \\ & / & \\ \hat{c}_1 & - & t_1(\hat{x}_2) \\ & \backslash & \\ & & f_1(\hat{x}_3) \end{array}$$

NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations

$$\bot_1 \;—\; \hat{x}_1 \;—\; \top_1 \qquad \bot_2 \;—\; \hat{x}_2 \;—\; \top_2 \qquad \bot_3 \;—\; \hat{x}_3 \;—\; \top_3$$

$\hat{c}_1$ — $t_1(\hat{x}_1)$, $\hat{c}_1$ — $t_1(\hat{x}_2)$, $\hat{c}_1$ — $f_1(\hat{x}_3)$

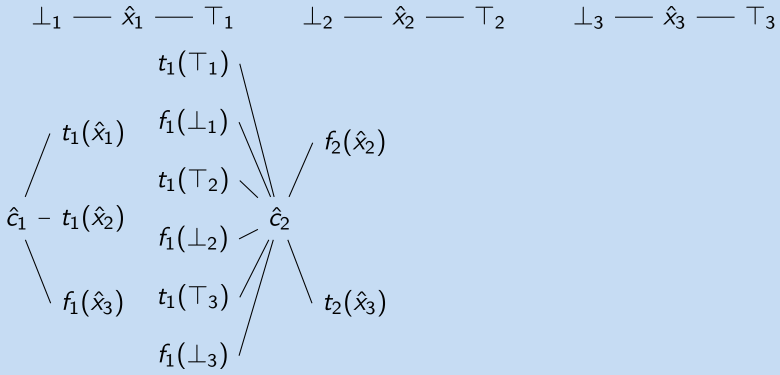
$t_1(\top_1)$ — $\hat{c}_2$, $f_1(\bot_1)$ — $\hat{c}_2$, $t_1(\top_2)$ — $\hat{c}_2$, $f_1(\bot_2)$ — $\hat{c}_2$, $t_1(\top_3)$ — $\hat{c}_2$, $f_1(\bot_3)$ — $\hat{c}_2$

NP-completeness proof sketch example

Formula

$$(x_1 \lor x_2 \lor \neg x_3) \land (\neg x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2)$$

Translation to equations

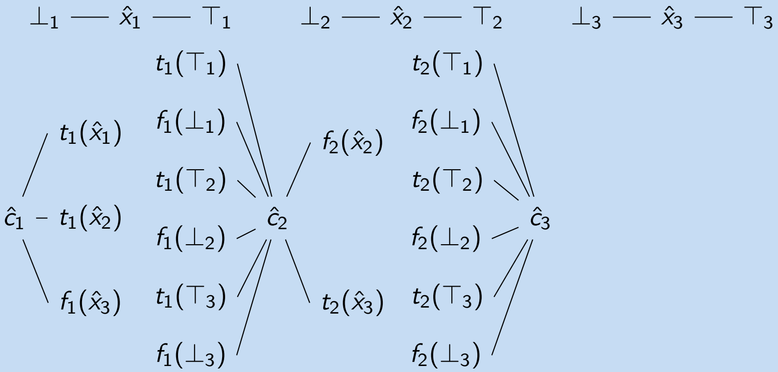


NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations

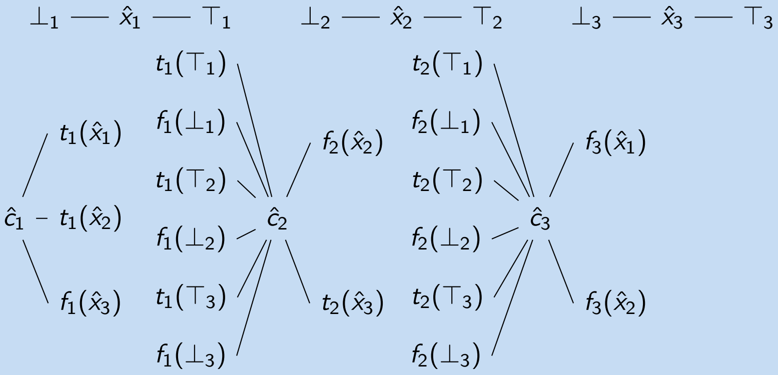


NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations

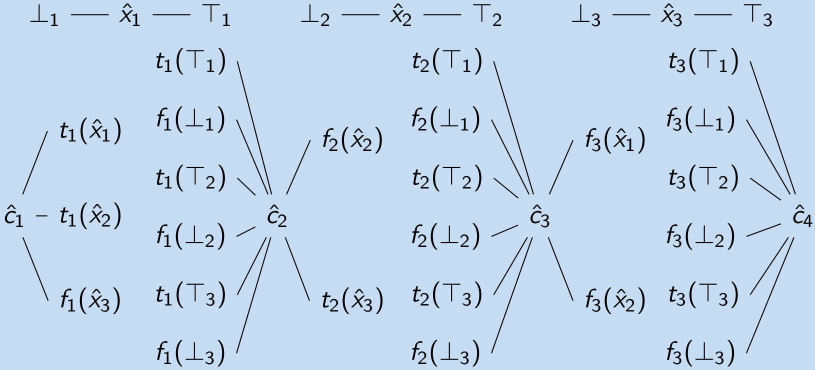


NP-completeness proof sketch example

Formula

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2)$$

Translation to equations



NP-completeness proof sketch example

Formula

$$(x_1 \lor x_2 \lor \neg x_3) \land (\neg x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2)$$

Small subset corresponding to satisfying assignment

$$\hat{x}_1 \text{ --- } \top_1 \qquad \bot_2 \text{ --- } \hat{x}_2 \qquad \hat{x}_3 \text{ --- } \top_3$$

$$\hat{c}_1 - t_1(\hat{x}_1) \quad t_1(\top_1) - \hat{c}_2 - f_2(\hat{x}_2) \quad f_2(\bot_2) - \hat{c}_3 - f_3(\hat{x}_2) \quad f_3(\bot_2) - \hat{c}_4$$

Space and Congruence Compression of Proofs

Is it possible to remove parts of the proof from memory during proof processing?

Is it possible to remove parts of the proof from memory during proof processing?

Which parts?

Space Measure Example

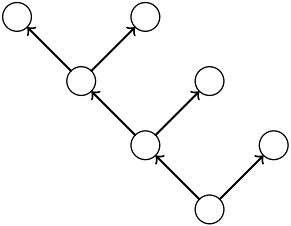
$$
\begin{tikzcd}[column sep=small]
x_1, x_2, \neg x_3 & & x_1, x_2 & & \\
& x_1, \neg x_3 \arrow[ul] \arrow[ur] & & x_1, x_3 & \\
& & x_1 \arrow[ul] \arrow[ur] & & \neg x_1 \\
& & & \bot \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Space Measure Example

Maximum number of nodes in memory: 0

○ Not in memory

● In memory

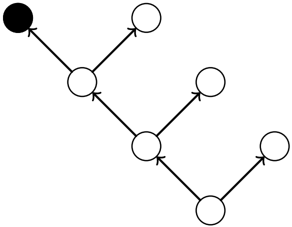


Space Measure Example

Maximum number of nodes in memory: 1

Not in memory

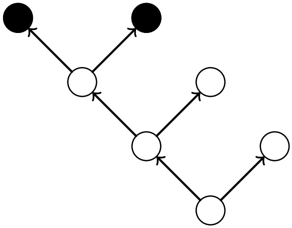
In memory



Maximum number of nodes in memory: 2


Not in memory


In memory

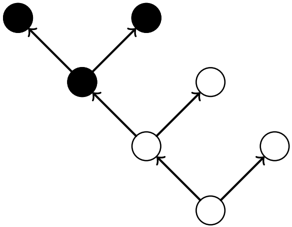


Space Measure Example

Maximum number of nodes in memory: 3

Not in memory

In memory

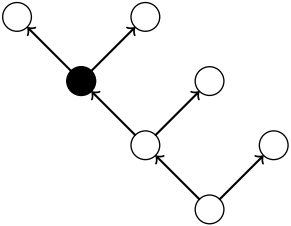


Space Measure Example

Maximum number of nodes in memory: 3

Not in memory

In memory

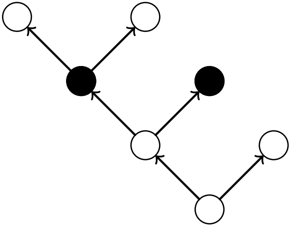


Space Measure Example

Maximum number of nodes in memory: 3

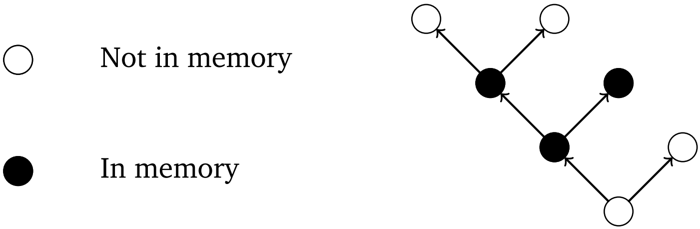
Not in memory

In memory



Space Measure Example

Maximum number of nodes in memory: 3

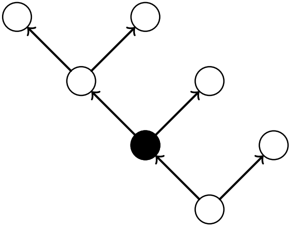


Space Measure Example

Maximum number of nodes in memory: 3

Not in memory

In memory

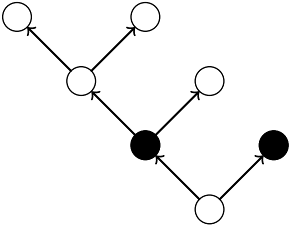


Space Measure Example

Maximum number of nodes in memory: 3

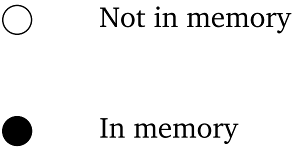
Not in memory

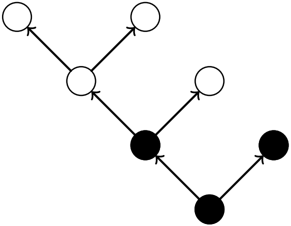
In memory



Space Measure Example

Maximum number of nodes in memory: 3



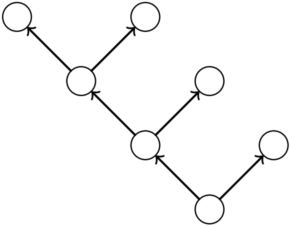


Space Measure Example

Maximum number of nodes in memory: 0

Not in memory

In memory

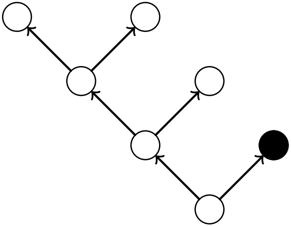


Space Measure Example

Maximum number of nodes in memory: 1

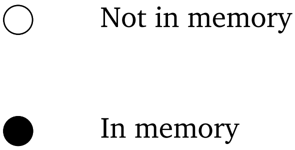
Not in memory

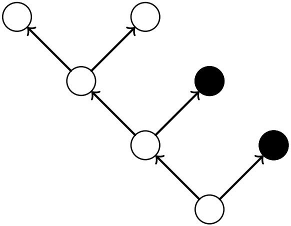
In memory



Space Measure Example

Maximum number of nodes in memory: 2



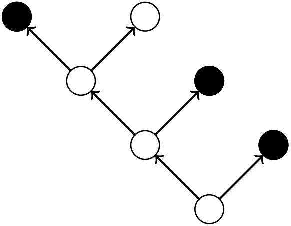


Space Measure Example

Maximum number of nodes in memory: 3

Not in memory

In memory

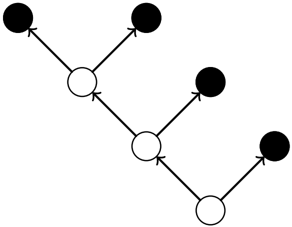


Space Measure Example

Maximum number of nodes in memory: 4

Not in memory

In memory

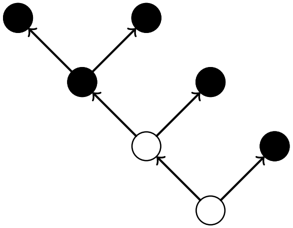


Space Measure Example

Maximum number of nodes in memory: 5

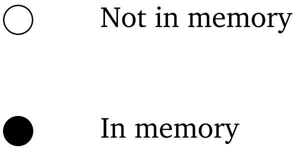
○ Not in memory

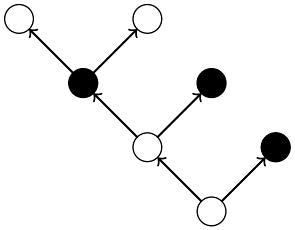
● In memory



Space Measure Example

Maximum number of nodes in memory: 5



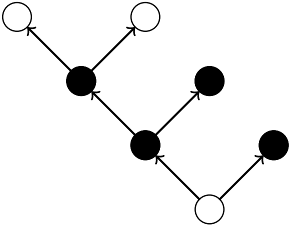


Space Measure Example

Maximum number of nodes in memory: 5

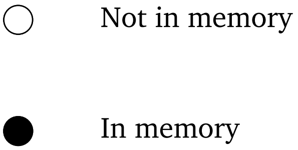
Not in memory

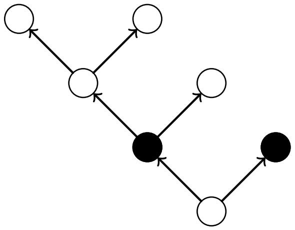
In memory



Space Measure Example

Maximum number of nodes in memory: 5

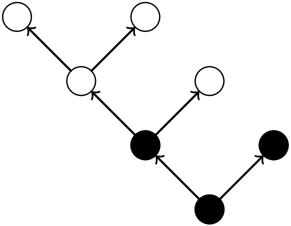




Maximum number of nodes in memory: 5

Not in memory

In memory

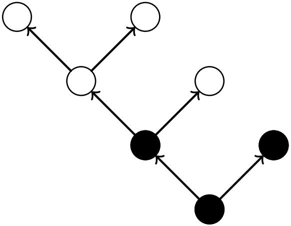


Space Measure Example

Maximum number of nodes in memory: 5

Not in memory

In memory



Good traversal orders are essential!

Space Measure

Space measure of a proof and a traversal order

Maximal amount of nodes that have to be kept in memory at once while processing the proof following the traversal order

Construct Traversal Orders

Construct Optimal Order

- NP-complete
- Optimal strategy in some pebbling game

Construct Good Order

- Greedy Algorithms
- Heuristic choices
- Top-Down
- Bottom-Up

Experiments, Unsung Heroes & Conclusion

Experimental Results

Congruence Compression

- 2% average effective compression in proof length
- 28% compression in explanation length

Space Compression

- Bottom-Up outperforms Top-Down
- Average space measure is 44.1 times smaller than proof length

Unsung Heroes

- Explanation producing congruence closure algorithm
  - Using immutable data structures
  - Modified version of Dijkstra's shortest path algorithm
- Proof producing algorithm
- Resolution calculus extended with equality
- SAT translation of optimal traversal order
- Correct- & soundness proofs
- Implementation of all presented methods

Conclusion

- Proofs can be compressed in length and space
- Finding the shortest explanation is NP-complete
- Proof production is tricky
- Construct traversal orders Bottom-Up

Thank you for your attention!

Short explanation, long proof

$$t_a \leftarrow f(f(a,b), f(a,a))$$

$$t_b \leftarrow f(f(b,a), f(b,b))$$

$$f(a,b) \neq f(b,a), f(a,a) \neq f(b,b), t_a = t_b$$

$$a \neq b, f(a,b) = f(b,a)$$

$$a \neq b, f(a,a) \neq f(b,b), t_a = t_b$$

$$a \neq b, f(a,a) = f(b,b)$$

$$a \neq b, t_a = t_b$$

Compatibility proof

$$t_a \neq a, a \neq b, b \neq t_b, t_a = t_b$$

Transitivity proof

Congruence Experimental Results

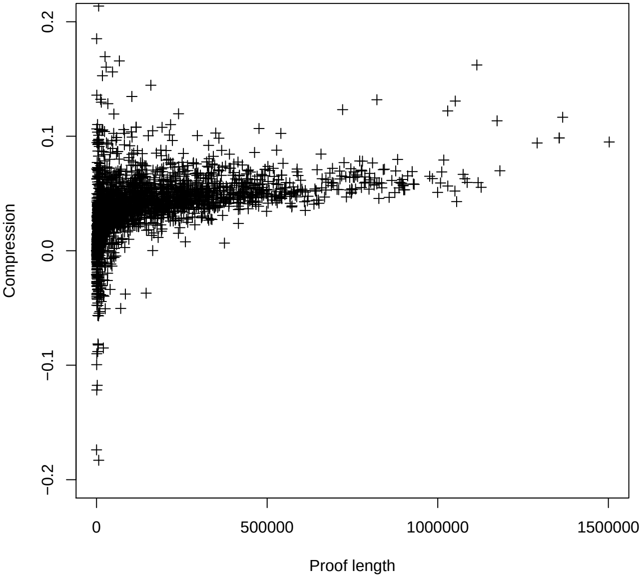
- 3965 proofs of problems of the SMT-LIB benchmark in the QF_UF logic

Method	Compression	Min	Max	Speed
EqGraph	**5.350** %	-18.302 %	**81.347** %	0.343
Proof Forest	5.196 %	-43.985 %	77.202 %	0.611
DAGify	3.368 %	**0.0** %	14.433 %	**1.655**

Table: Compression Results

Congruence Graph	Compressed	Compression
Equation Graph	12.42 %	28.34 %
Proof Forest	11.459 %	28.69 %

Table: Explantion Size Results


Compression
0.2
0.1
0.0
−0.1
−0.2
0
500000
1000000
1500000
Proof length

Space Compression Results

- VeriT: Problems from SMT-lib
- TraceCheck: Problems from SATLIB, computed with PicoSAT

Name	Number of proofs	Maximum length	Average length
TraceCheck_1	2239	90756	5423
TraceCheck_2	215	1768249	268863
veriT_1	4187	2241042	103162
veriT_2	914	120075	5391

Table: Proof Benchmark Sets

Space Compression Results

Algorithm Heuristic	**Relative Performance** (%)	**Speed** (nodes/ms)
Bottom-Up		
Children	17.52	**88.6**
LastChild	**26.31**	84.5
Distance(1)	9.46	21.2
Distance(3)	-0.40	0.5
Top-Down		
Children	-27.47	0.3
LastChild	-31.98	1.9
Distance(1)	-70.14	0.6
Distance(3)	**-74.33**	**0.1**

Table: Experimental Results

Construct Order Top-Down

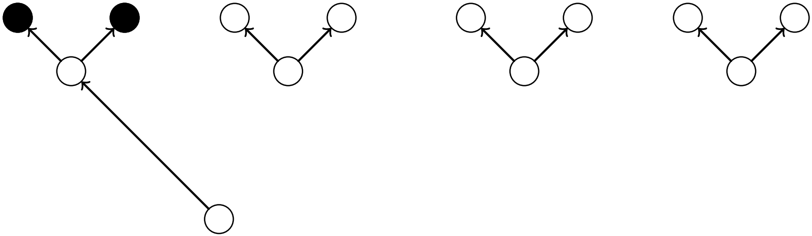
Maximum number of nodes in memory: 0

Construct Order Top-Down

Maximum number of nodes in memory: 1

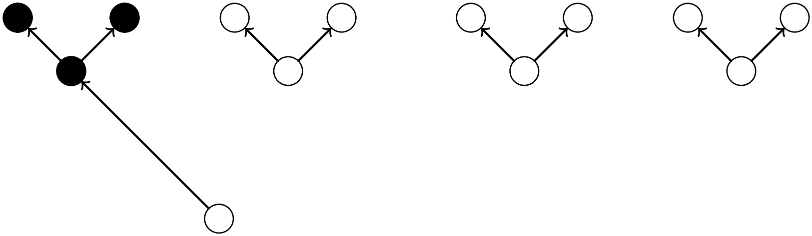
Construct Order Top-Down

Maximum number of nodes in memory: 2



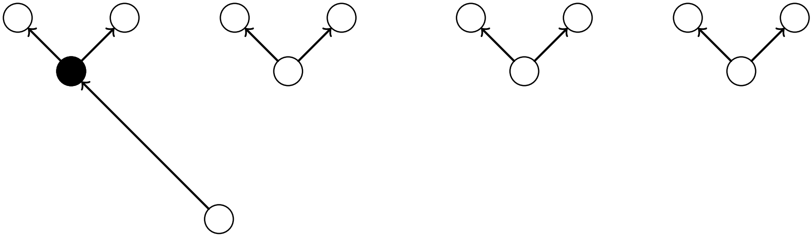
Construct Order Top-Down

Maximum number of nodes in memory: 3



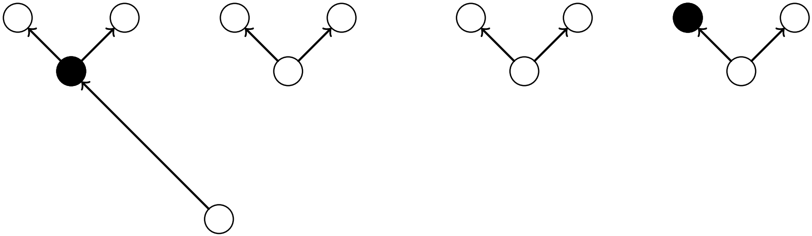
Construct Order Top-Down

Maximum number of nodes in memory: 3



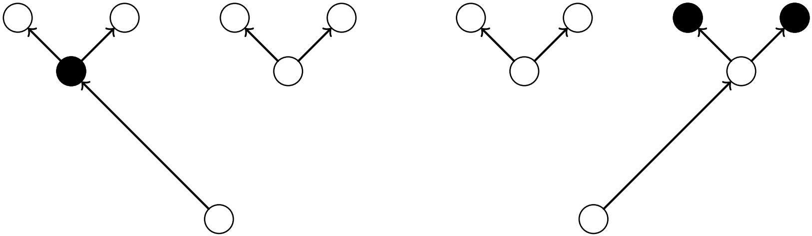
Construct Order Top-Down

Maximum number of nodes in memory: 3



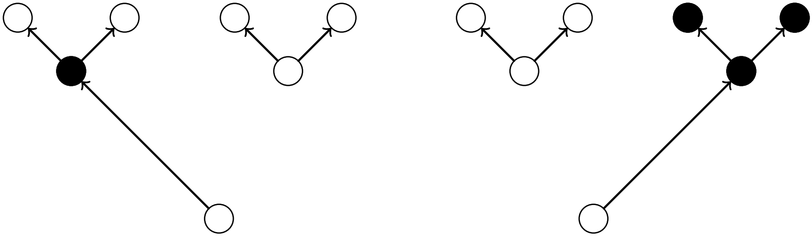
Construct Order Top-Down

Maximum number of nodes in memory: 3



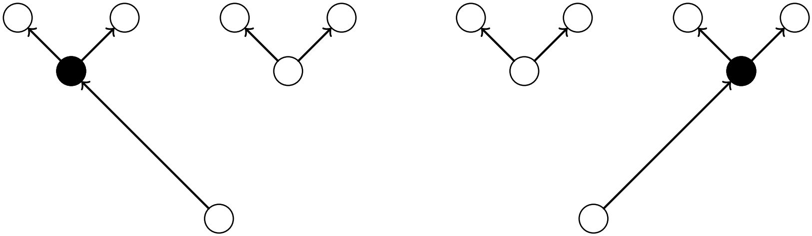
Construct Order Top-Down

Maximum number of nodes in memory: 4



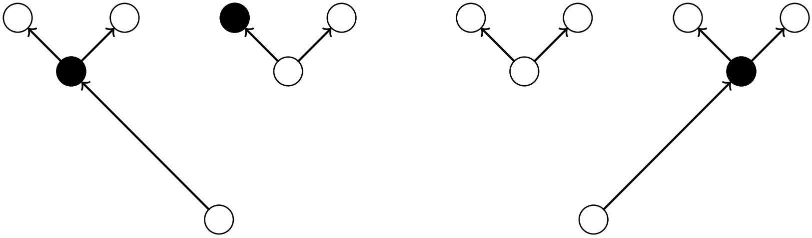
Construct Order Top-Down

Maximum number of nodes in memory: 4



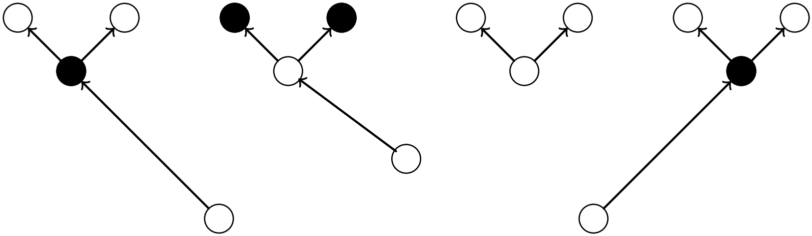
Construct Order Top-Down

Maximum number of nodes in memory: 4



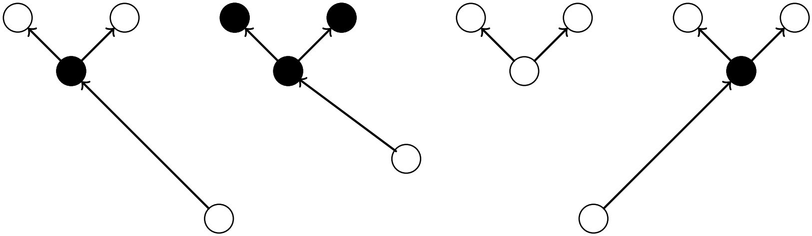
Construct Order Top-Down

Maximum number of nodes in memory: 4



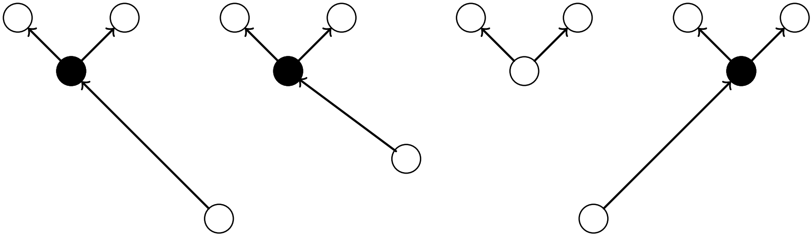
Construct Order Top-Down

Maximum number of nodes in memory: 5



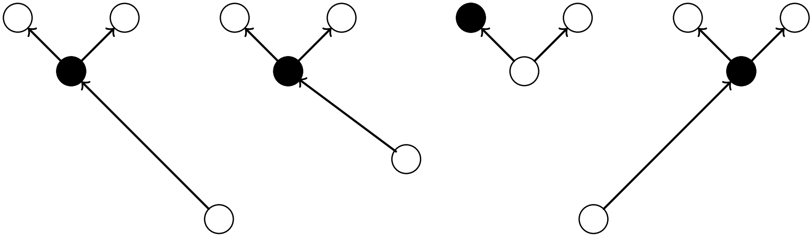
Construct Order Top-Down

Maximum number of nodes in memory: 5



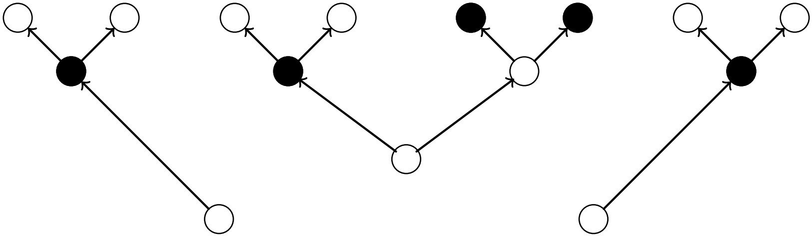
Construct Order Top-Down

Maximum number of nodes in memory: 5



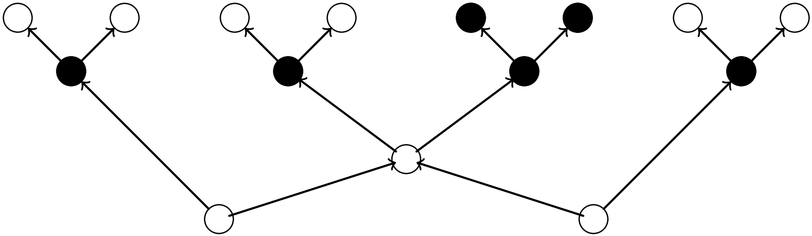
Construct Order Top-Down

Maximum number of nodes in memory: 5



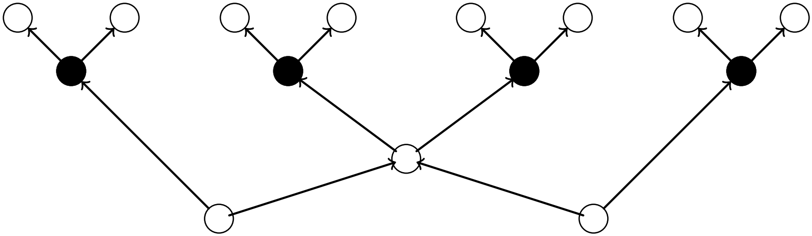
Construct Order Top-Down

Maximum number of nodes in memory: 6



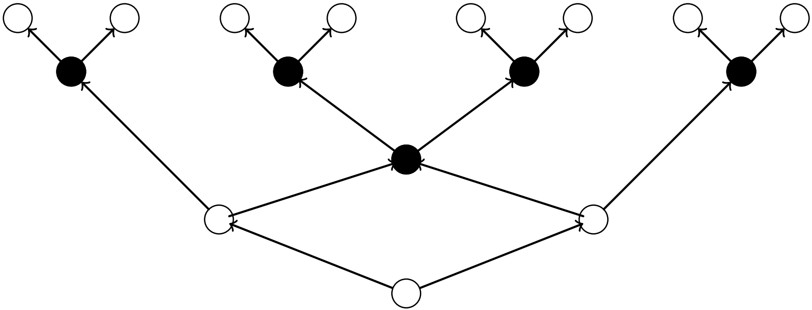
Construct Order Top-Down

Maximum number of nodes in memory: 6



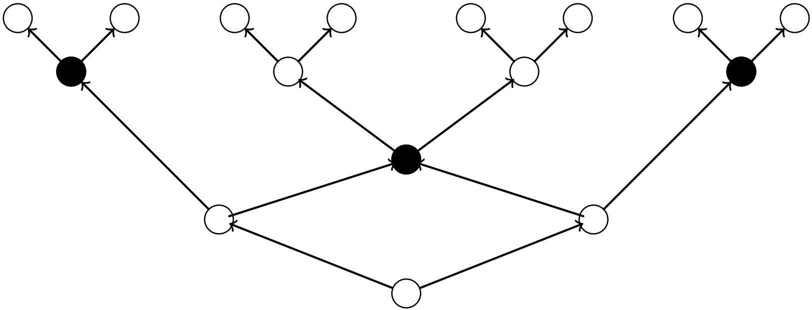
Construct Order Top-Down

Maximum number of nodes in memory: 6



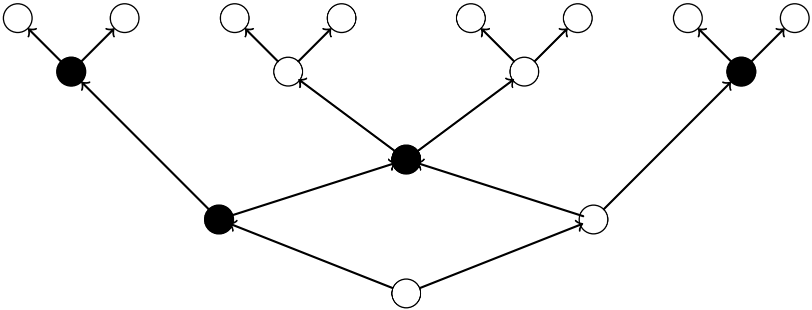
Construct Order Top-Down

Maximum number of nodes in memory: 6



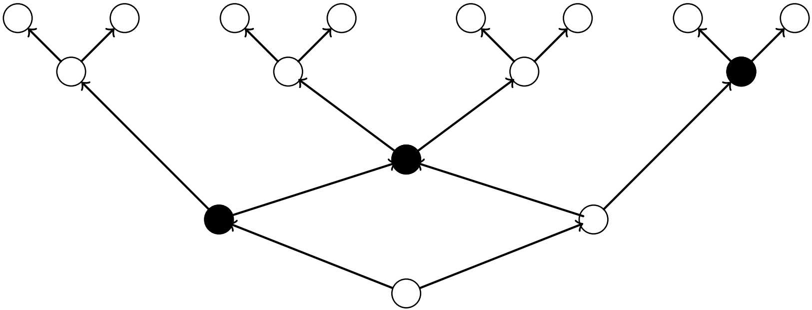
Construct Order Top-Down

Maximum number of nodes in memory: 6



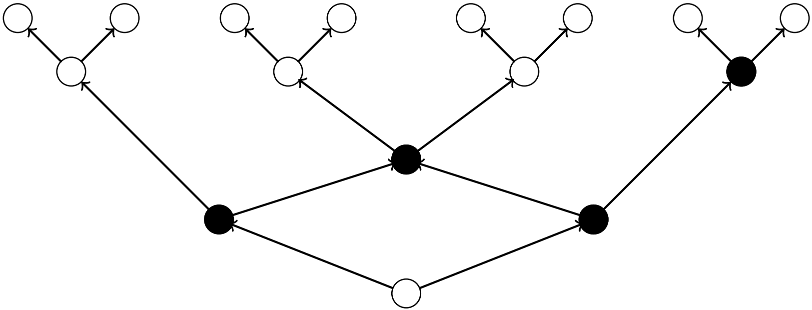
Construct Order Top-Down

Maximum number of nodes in memory: 6



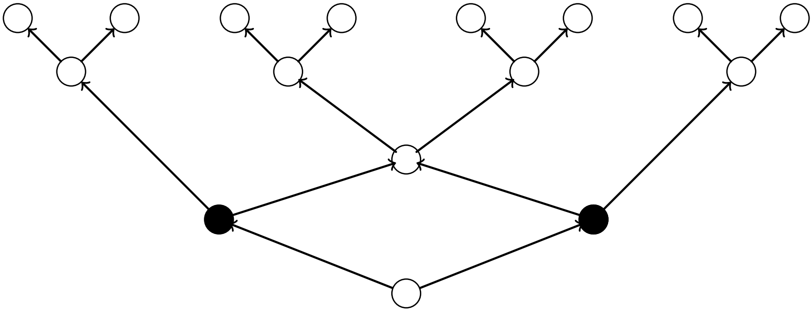
Construct Order Top-Down

Maximum number of nodes in memory: 6



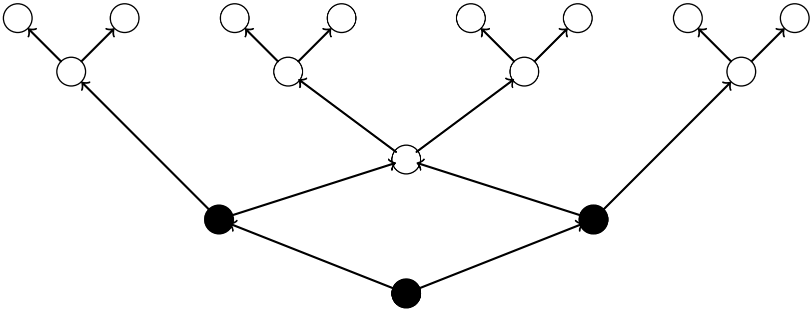
Construct Order Top-Down

Maximum number of nodes in memory: 6



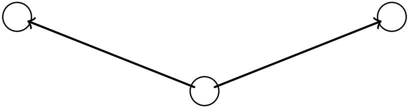
Construct Order Top-Down

Maximum number of nodes in memory: 6



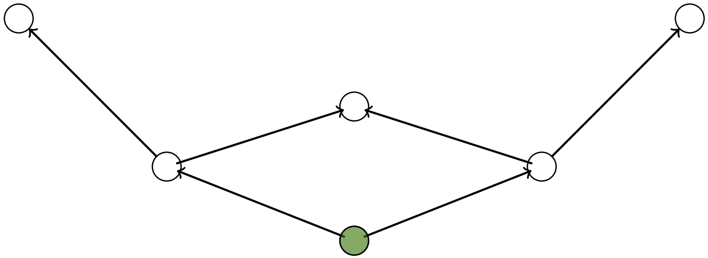
Construct Order Bottom-Up

Maximum number of nodes in memory: 0



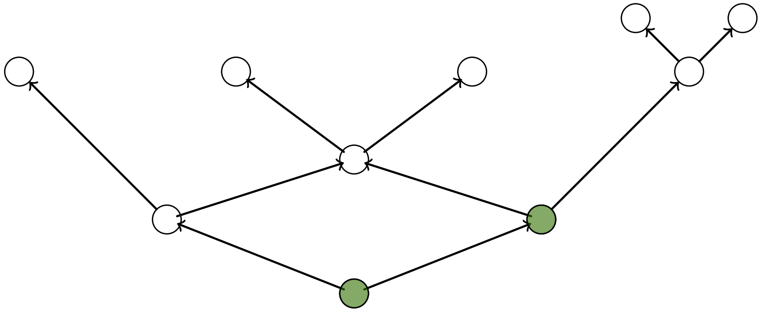
Construct Order Bottom-Up

Maximum number of nodes in memory: 0



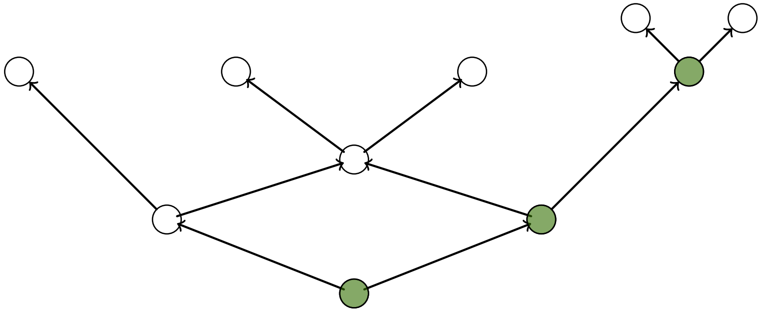
Construct Order Bottom-Up

Maximum number of nodes in memory: 0



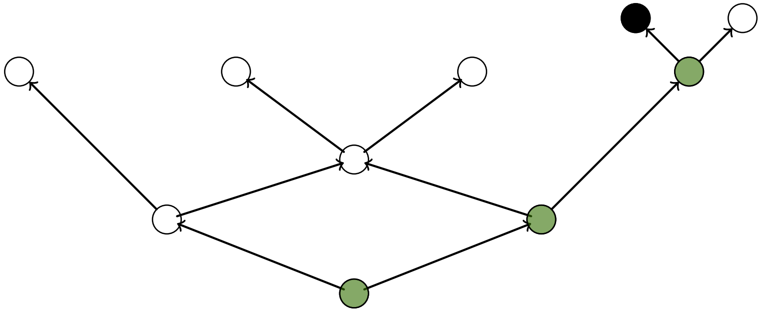
Construct Order Bottom-Up

Maximum number of nodes in memory: 0



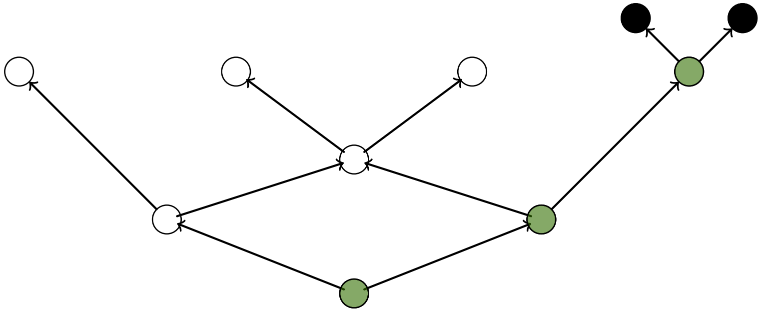
Construct Order Bottom-Up

Maximum number of nodes in memory: 1



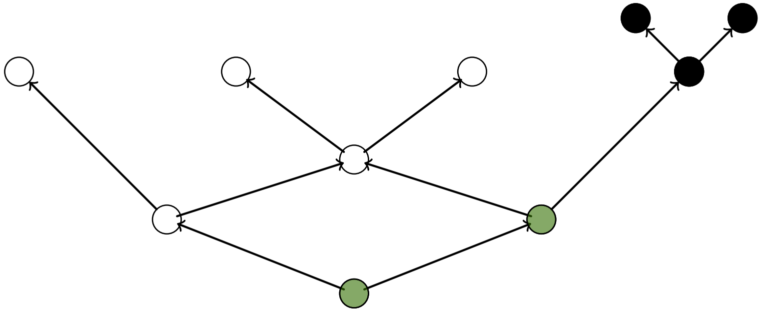
Construct Order Bottom-Up

Maximum number of nodes in memory: 2



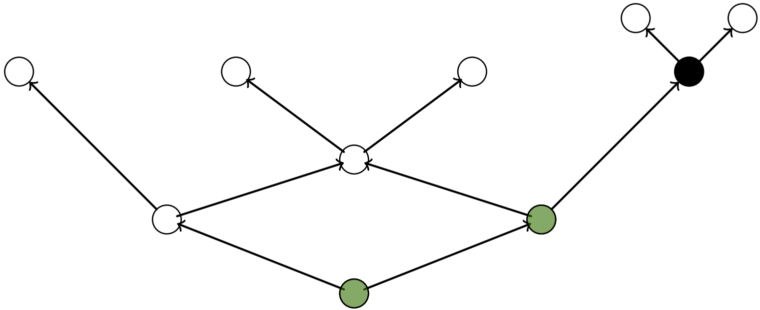
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



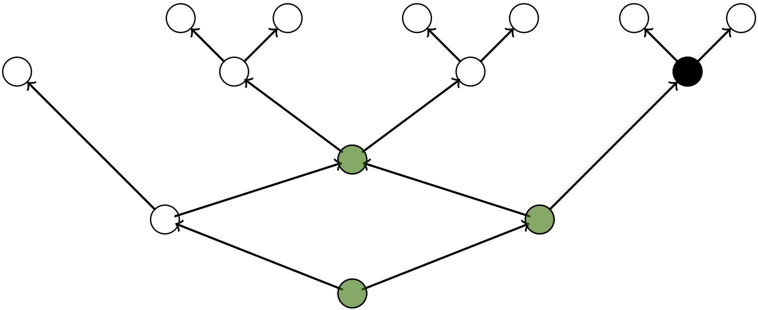
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



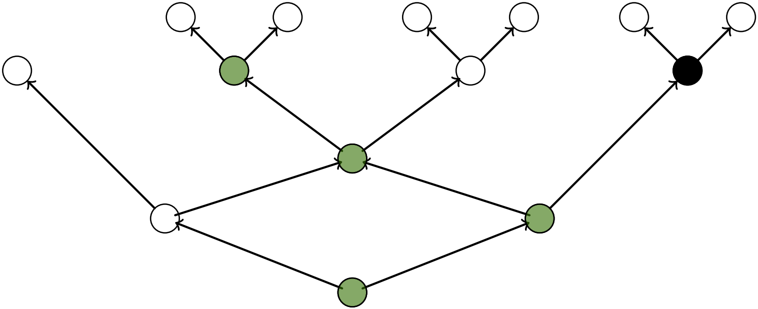
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



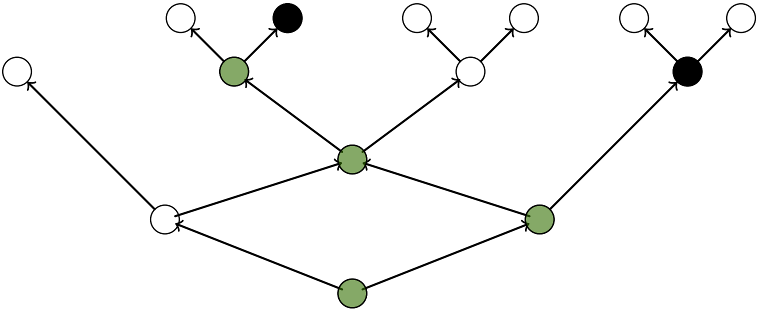
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



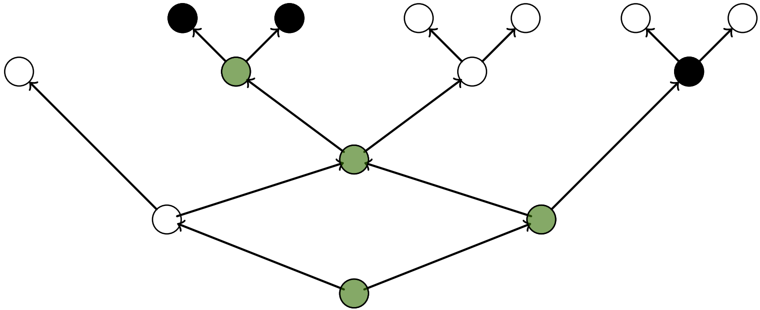
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



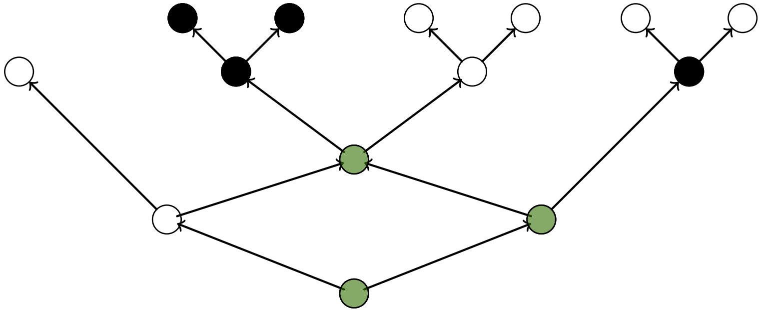
Construct Order Bottom-Up

Maximum number of nodes in memory: 3



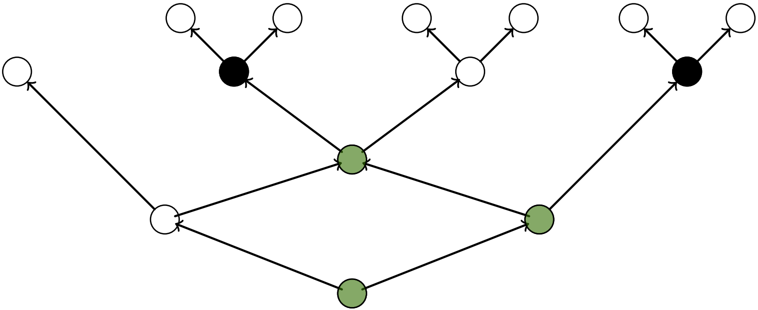
Construct Order Bottom-Up

Maximum number of nodes in memory: 4



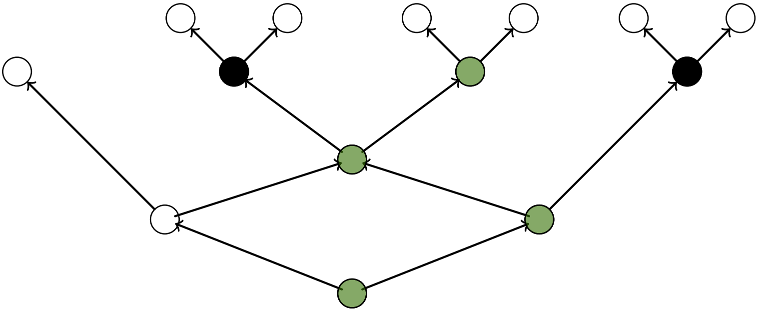
Construct Order Bottom-Up

Maximum number of nodes in memory: 4



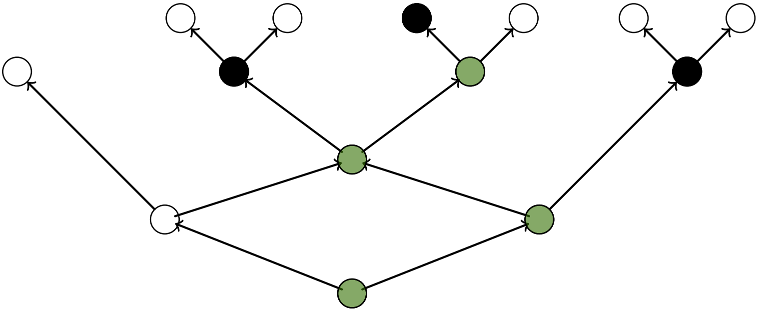
Construct Order Bottom-Up

Maximum number of nodes in memory: 4



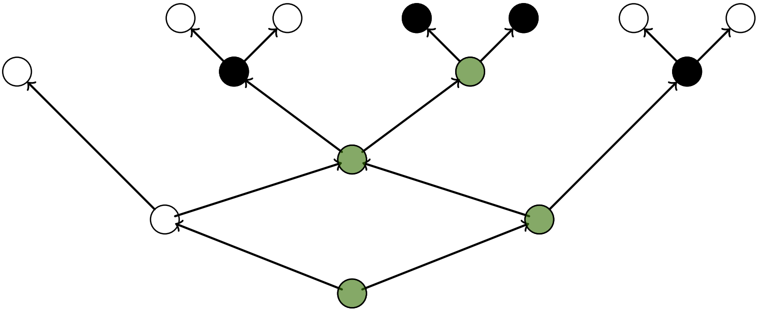
Construct Order Bottom-Up

Maximum number of nodes in memory: 4



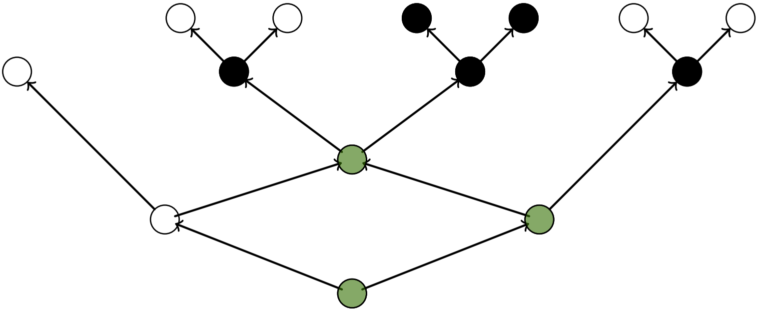
Construct Order Bottom-Up

Maximum number of nodes in memory: 4



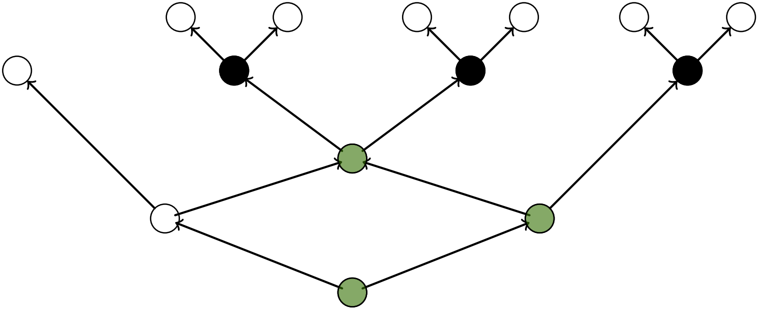
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



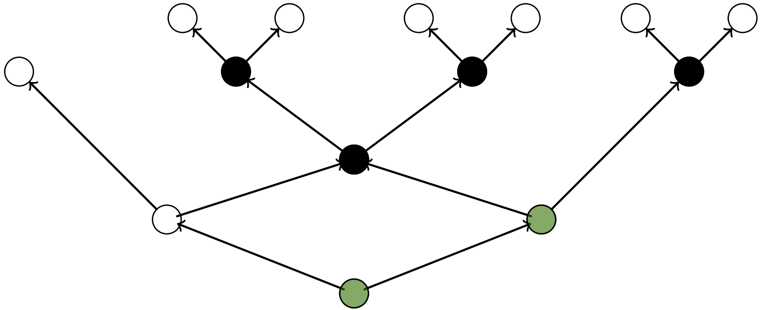
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



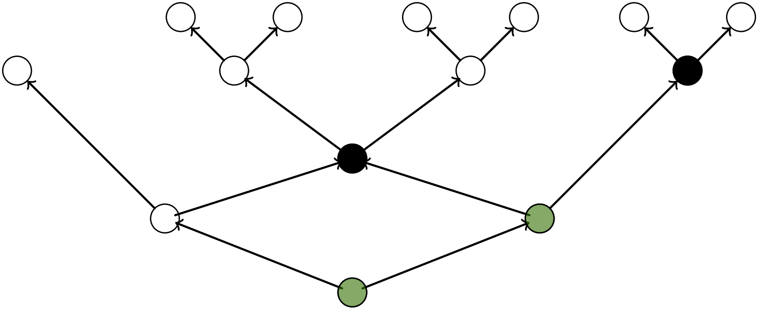
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



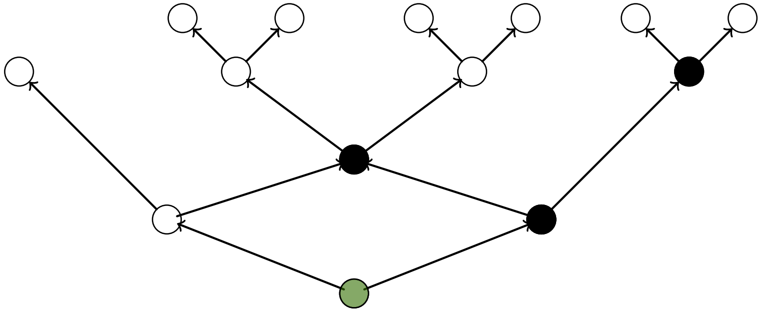
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



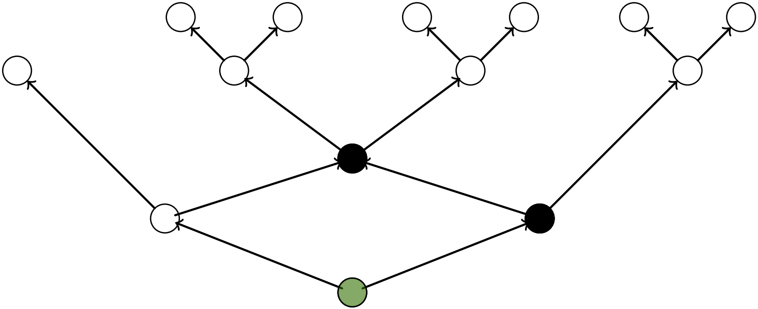
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



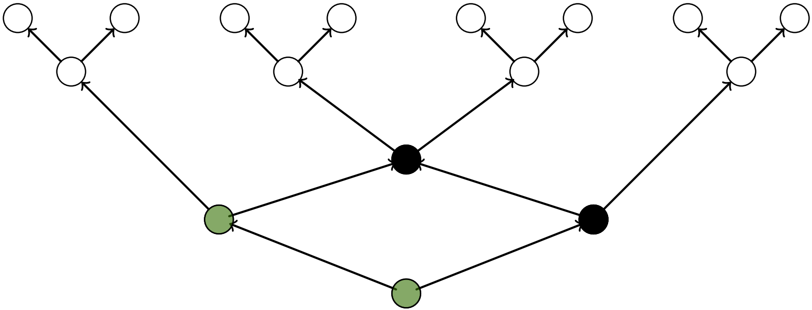
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



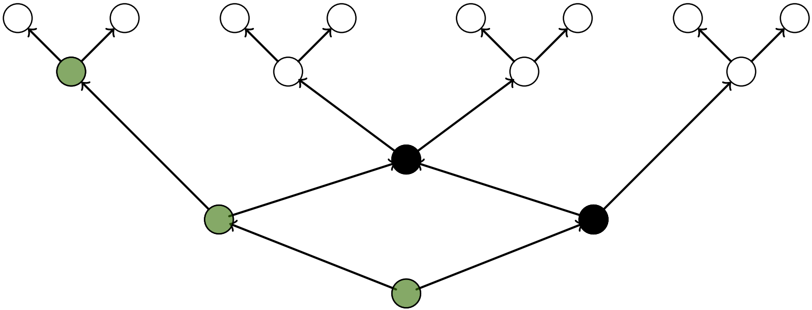
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



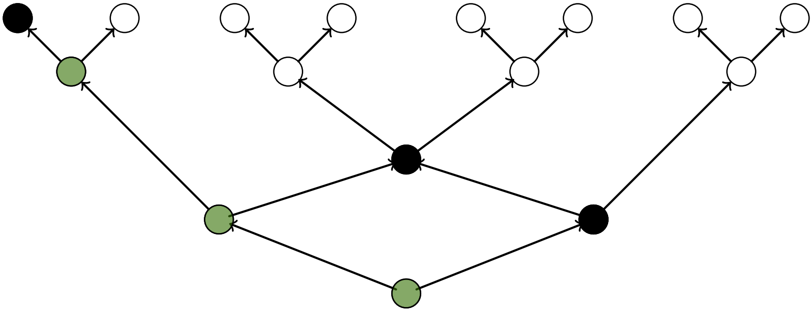
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



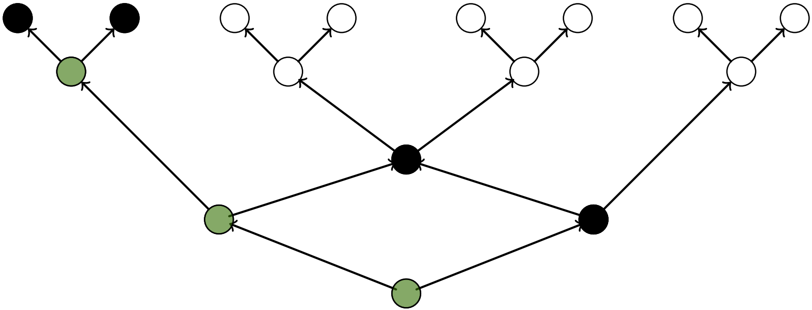
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



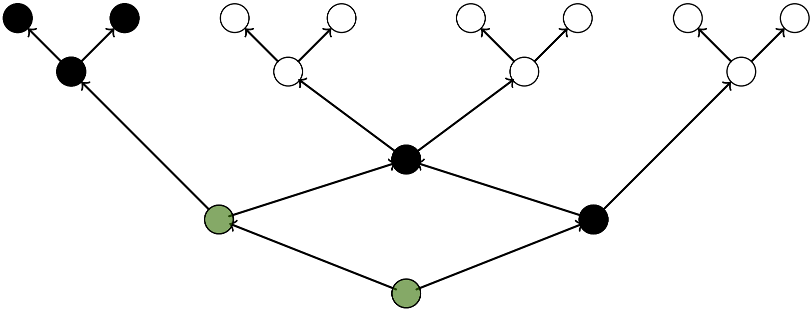
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



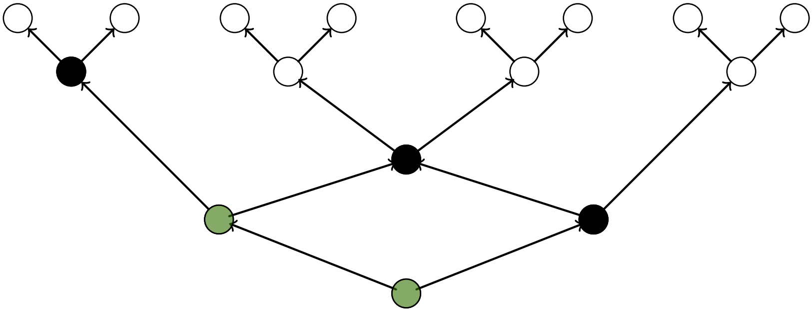
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



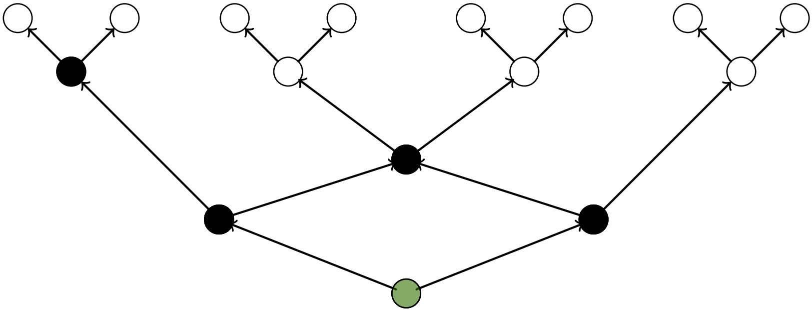
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



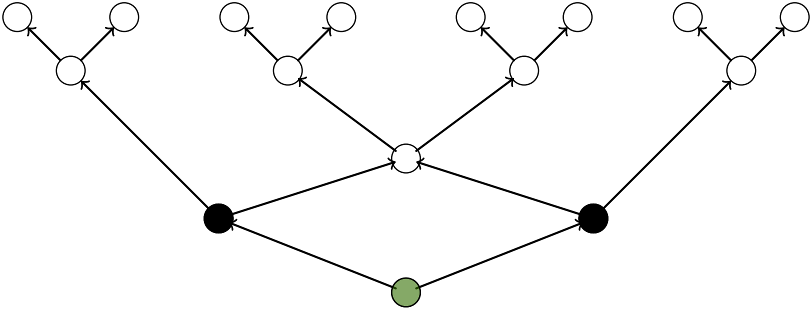
Construct Order Bottom-Up

Maximum number of nodes in memory: 5



Construct Order Bottom-Up

Maximum number of nodes in memory: 5



Construct Order Bottom-Up

Maximum number of nodes in memory: 5