Scaling SAT/MaxSAT encodings to large instances with SLIM

TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

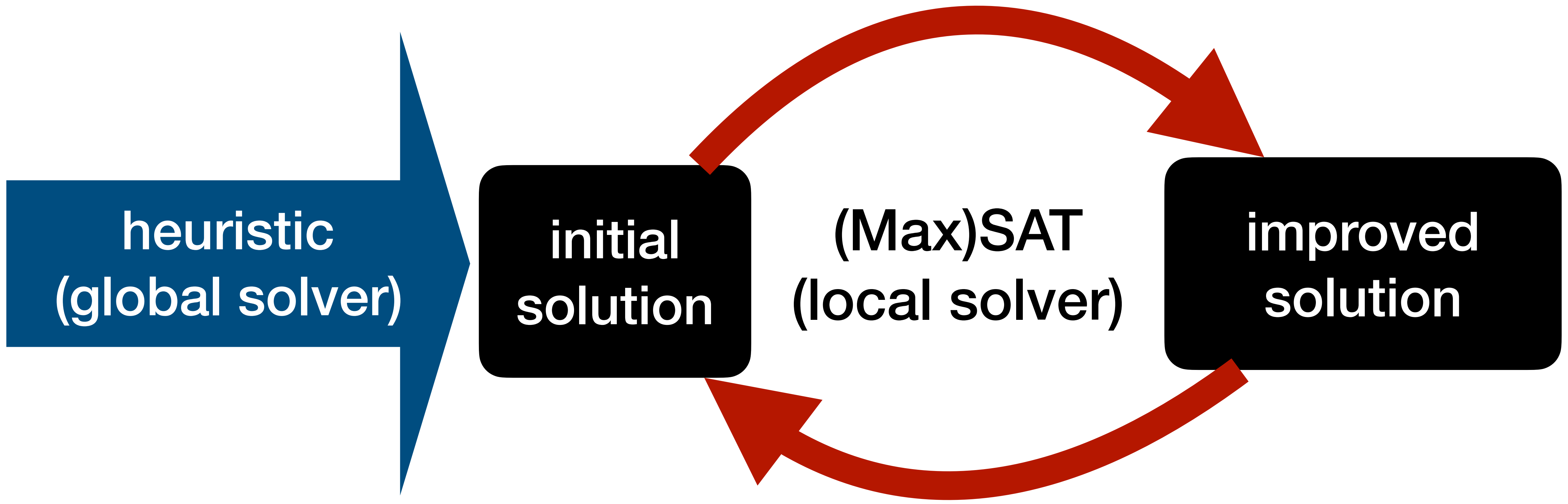
ALGORITHMS AND COMPLEXITY GROUP

Stefan Szeider

Motivation

- SAT encodings often blow up the instance size by a polynomial of order 3 or 4
- $60^4 > 12$ Mio
- Hence there is a certain hard limit on instance size for SAT encodings
- **S**AT-based **L**ocal **I**mprove **M**ethod (SLIM) tries to overcome this limit
- Idea: use a heuristic to compute an initial solution, then repeatedly apply SAT-encodings (or Max-SAT encodings) to local parts, to improve the initial solution
- SLIM is related to the LNS meta-heuristic, but SLIM defines the neighbourhood in a very structured way

The SLIM Loop



- **global solver**
- **local Solver**
- local **selection strategy**
- **budget**: size of the local instance
- **local timeout**: time allotted for the local solver
- **challenge**: ensure that the new local solution fits into the global solution

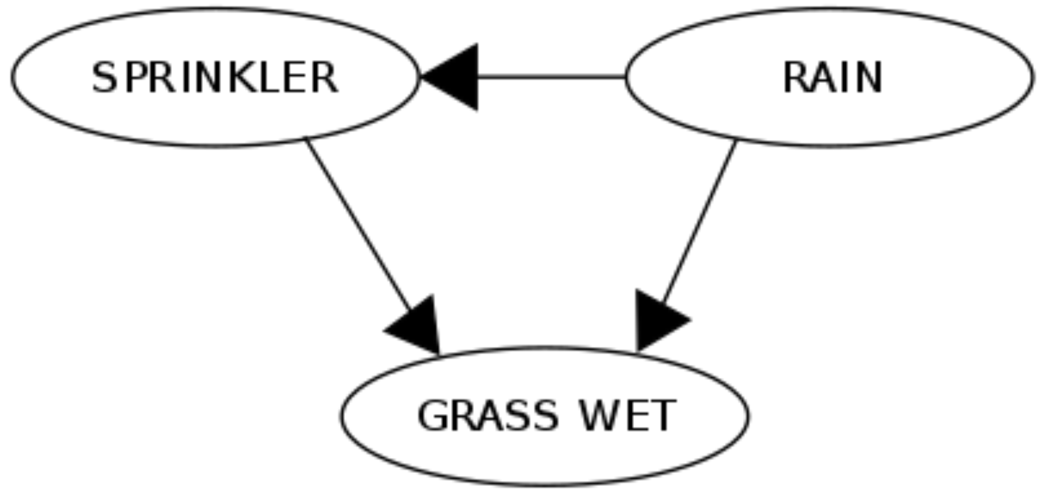
SLIM Showcases

Problem	local solver	Paper
Branchwidth	SAT	[Lodha, Ordyniak, Sz. (SAT'17, ToCL'19)]
Treewidth	SAT	[Fichte, Lodha, Sz. (SAT'17)]
Treedepth	MaxSAT	[Peruvemba Ramaswamy, Sz (CP'20)]
BN Structure Learning	MaxSAT	[Peruvemba Ramaswamy, Sz (AAAI'21)]
Decision Trees	SAT	[Schidler, Sz. (AAAI'21)]

Bayesian Network Structure Learning

Bayesian Network Structure Learning

	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99

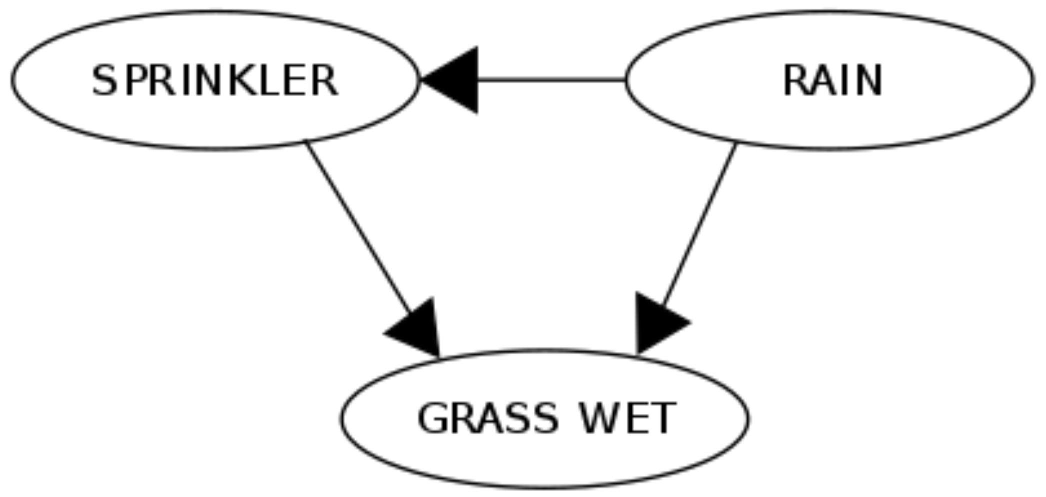


RAIN	
T	F
0.2	0.8

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Bayesian Network Structure Learning

RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99



RAIN T	RAIN F
0.2	0.8

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

- Reasoning on BNs is #P-complete
- but fixed-parameter tractable in the treewidth of the network's moral graph
- one of the few examples outside theory where treewidth is actually used

Score-Based Structure Learning

Sample data

a	b	c	d	e	f
1	0	0	1	1	0
1	1	1	1	1	1
0	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	1
...					

Score-Based Structure Learning

Sample data

a	b	c	d	e	f
1	0	0	1	1	0
1	1	1	1	1	1
0	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	1
...					

node	parent set	score
a	∅	0.12
a	{b}	0.1
a	{a,b}	0
b	∅	0.11
b	{a}	0.27
b	{a,c}	0.33
c	∅	0.01
c	{b}	0.33
c	{a,b}	0.45
...		

Score function cache

Score-Based Structure Learning

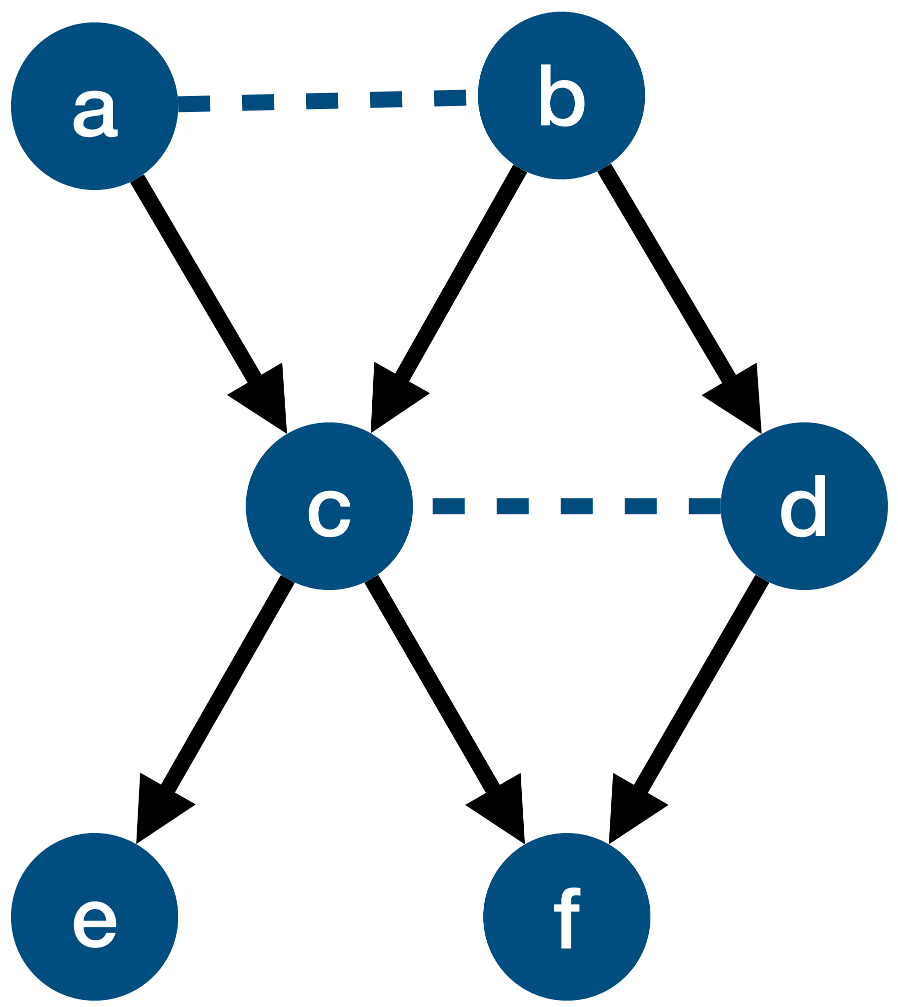
Sample data

a	b	c	d	e	f
1	0	0	1	1	0
1	1	1	1	1	1
0	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	1
...					

node	parent set	score
a	$\emptyset$	0.12
a	{b}	0.1
a	{a,b}	0
b	$\emptyset$	0.11
b	{a}	0.27
b	{a,c}	0.33
c	$\emptyset$	0.01
c	{b}	0.33
c	{a,b}	0.45
...		

Score function cache

BN DAG
+ moral edges



Score-Based Structure Learning

Sample data

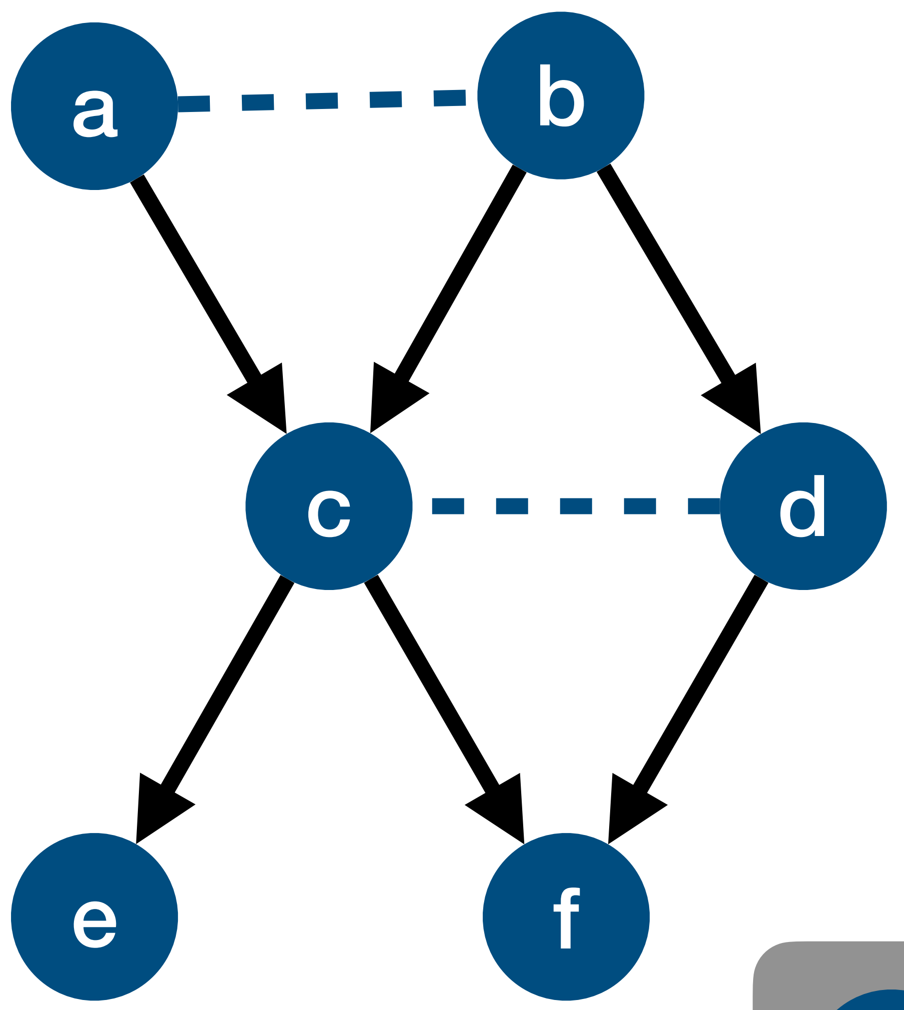
a	b	c	d	e	f
1	0	0	1	1	0
1	1	1	1	1	1
0	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	1
...					

node	parent set	score
a	∅	0.12
a	{b}	0.1
a	{a,b}	0
b	∅	0.11
b	{a}	0.27
b	{a,c}	0.33
c	∅	0.01
c	{b}	0.33
c	{a,b}	0.45
...		

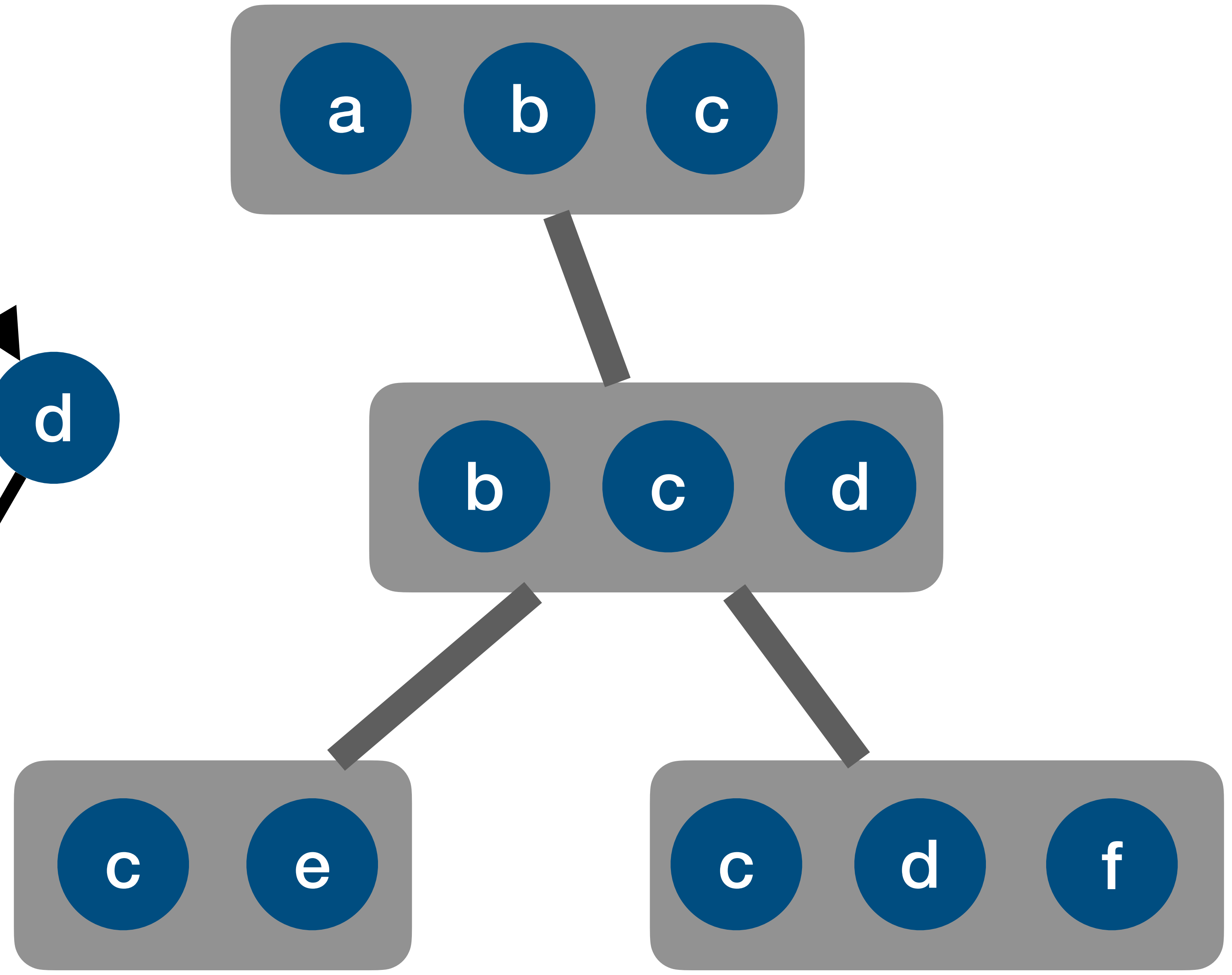
Score function cache

BN DAG

+ moral edges



tree decomposition width 2



Score-Based Structure Learning

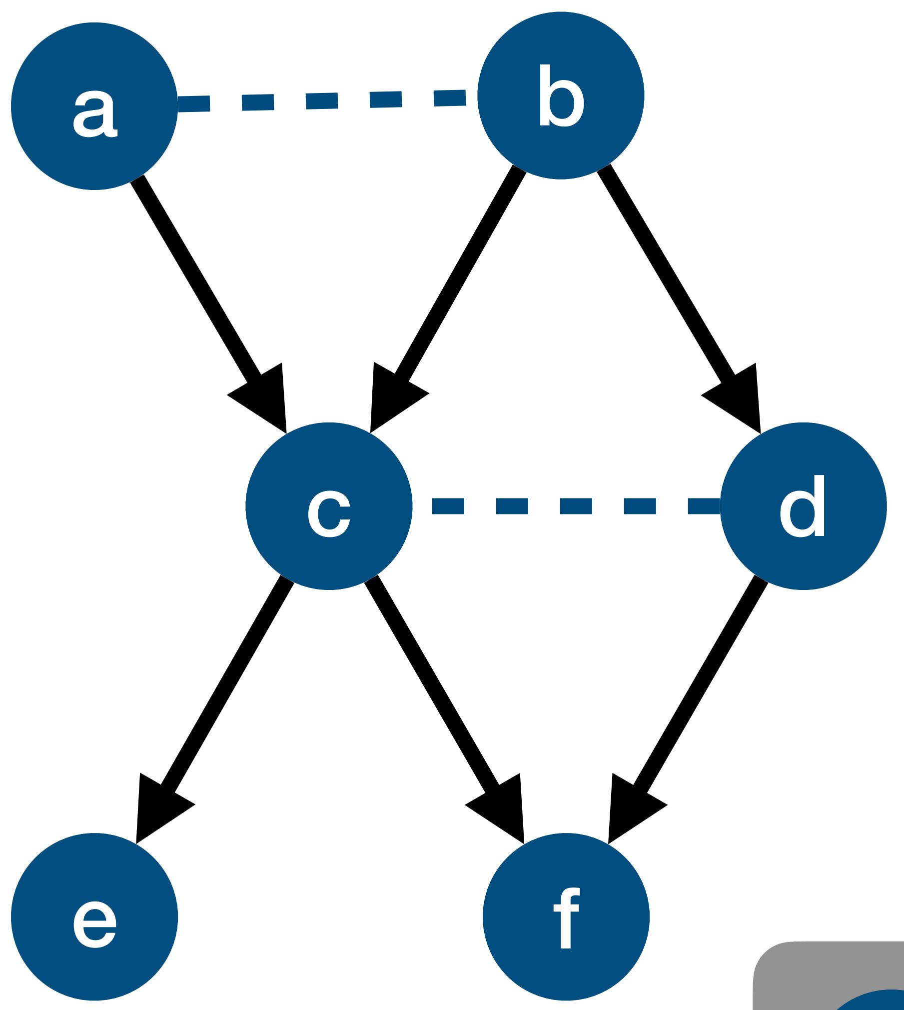
Sample data

a	b	c	d	e	f
1	0	0	1	1	0
1	1	1	1	1	1
0	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	1	0	1
1	0	0	1	0	1
...					

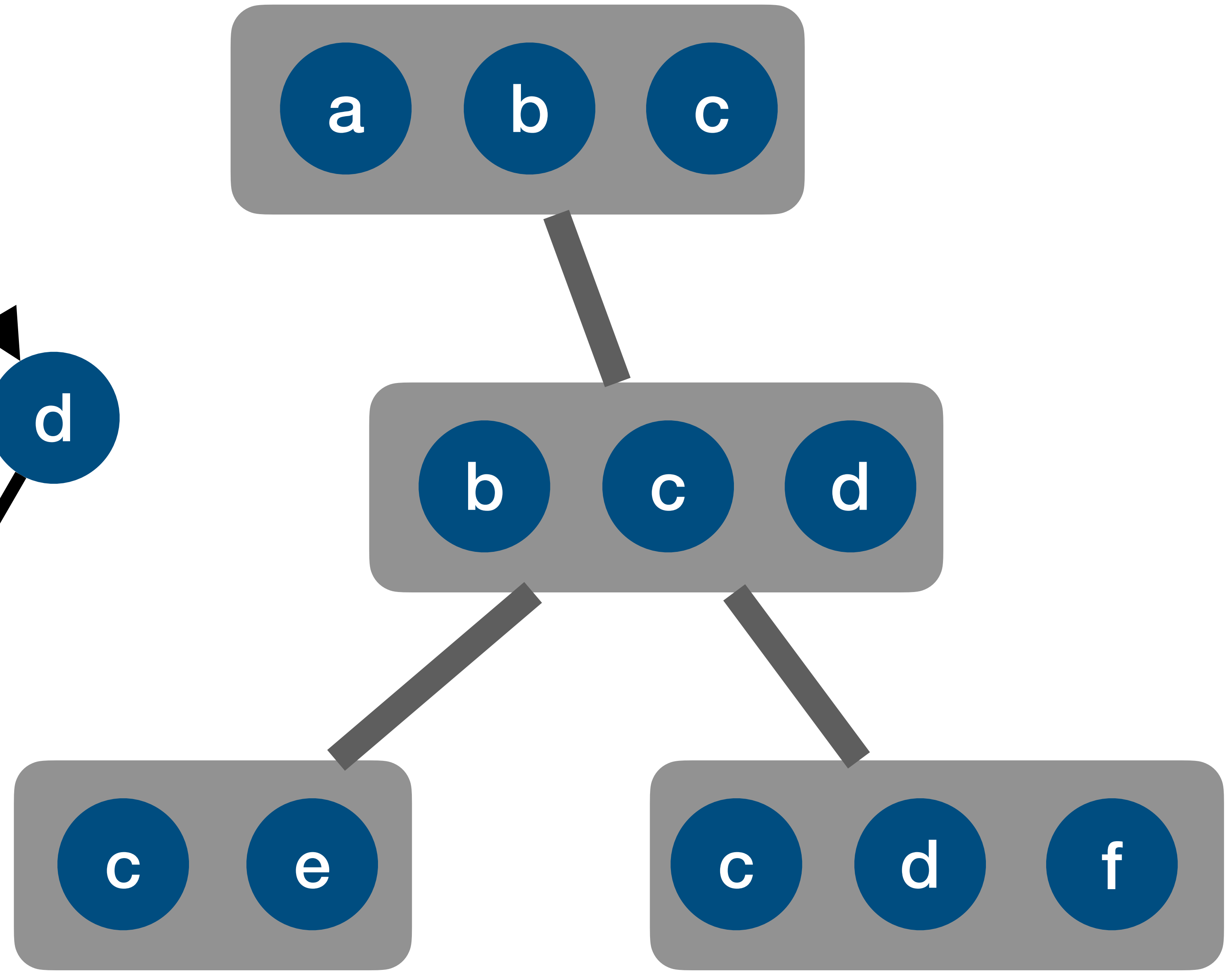
node	parent set	score
a	∅	0.12
a	{b}	0.1
a	{a,b}	0
b	∅	0.11
b	{a}	0.27
b	{a,c}	0.33
c	∅	0.01
c	{b}	0.33
c	{a,b}	0.45
...		

Score function cache

BN DAG
+ moral edges



tree decomposition
width 2

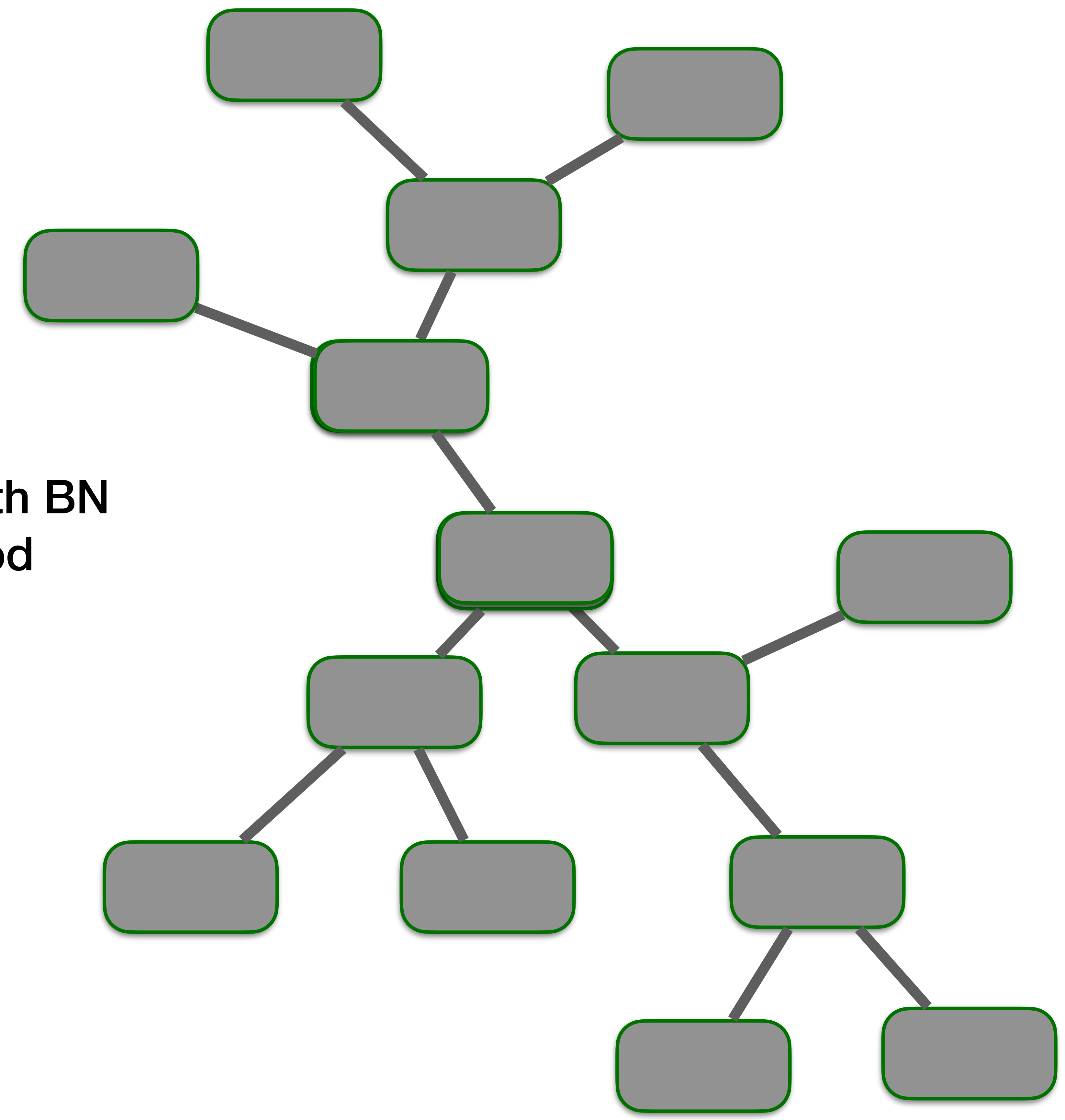


Hard clauses for acyclicity, bounded treewidth

Soft clauses for score maximisation

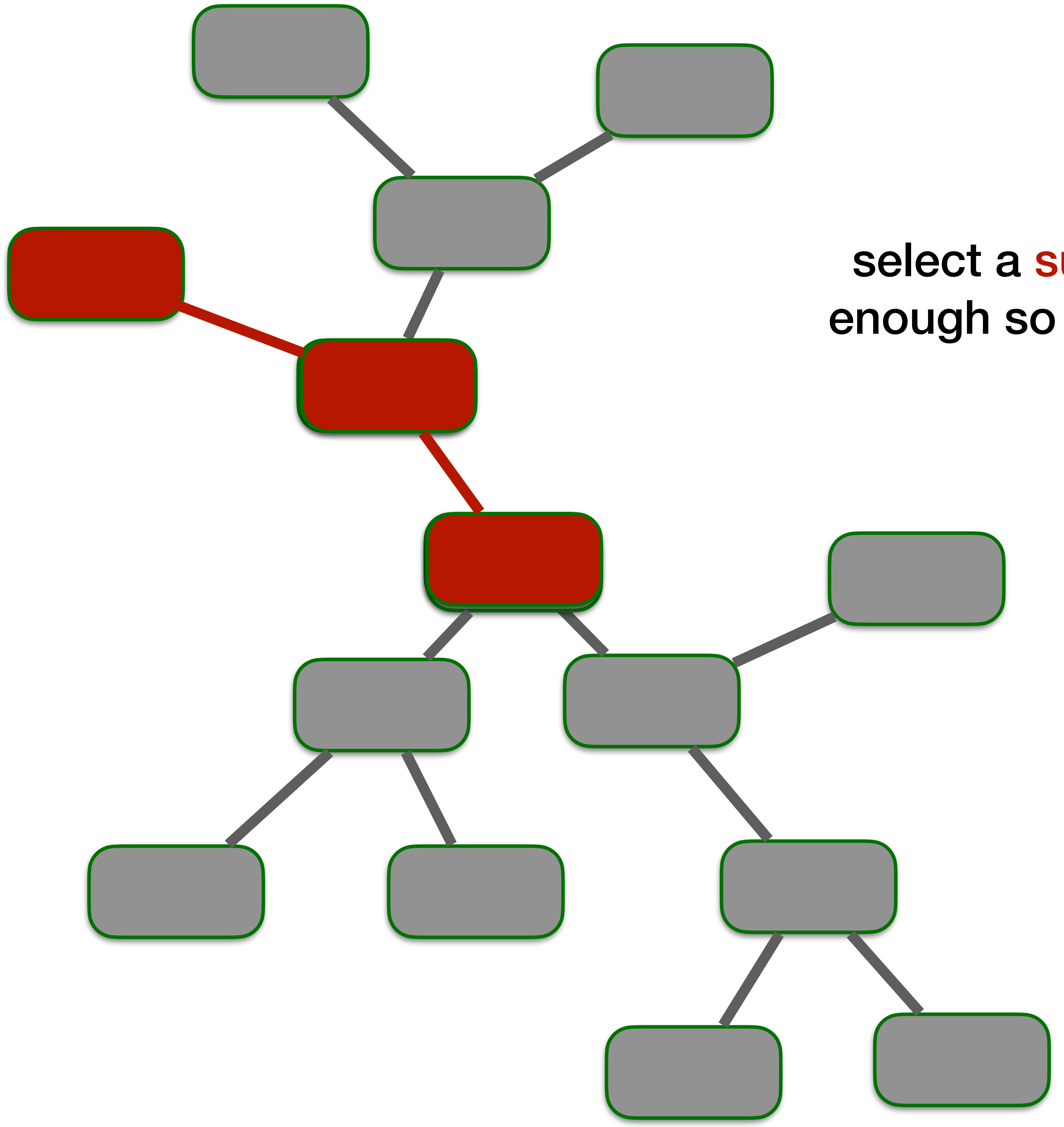
Partial Max-SAT

SAT-Based Local Improvement for BN Structure Learning (BN-SLIM)



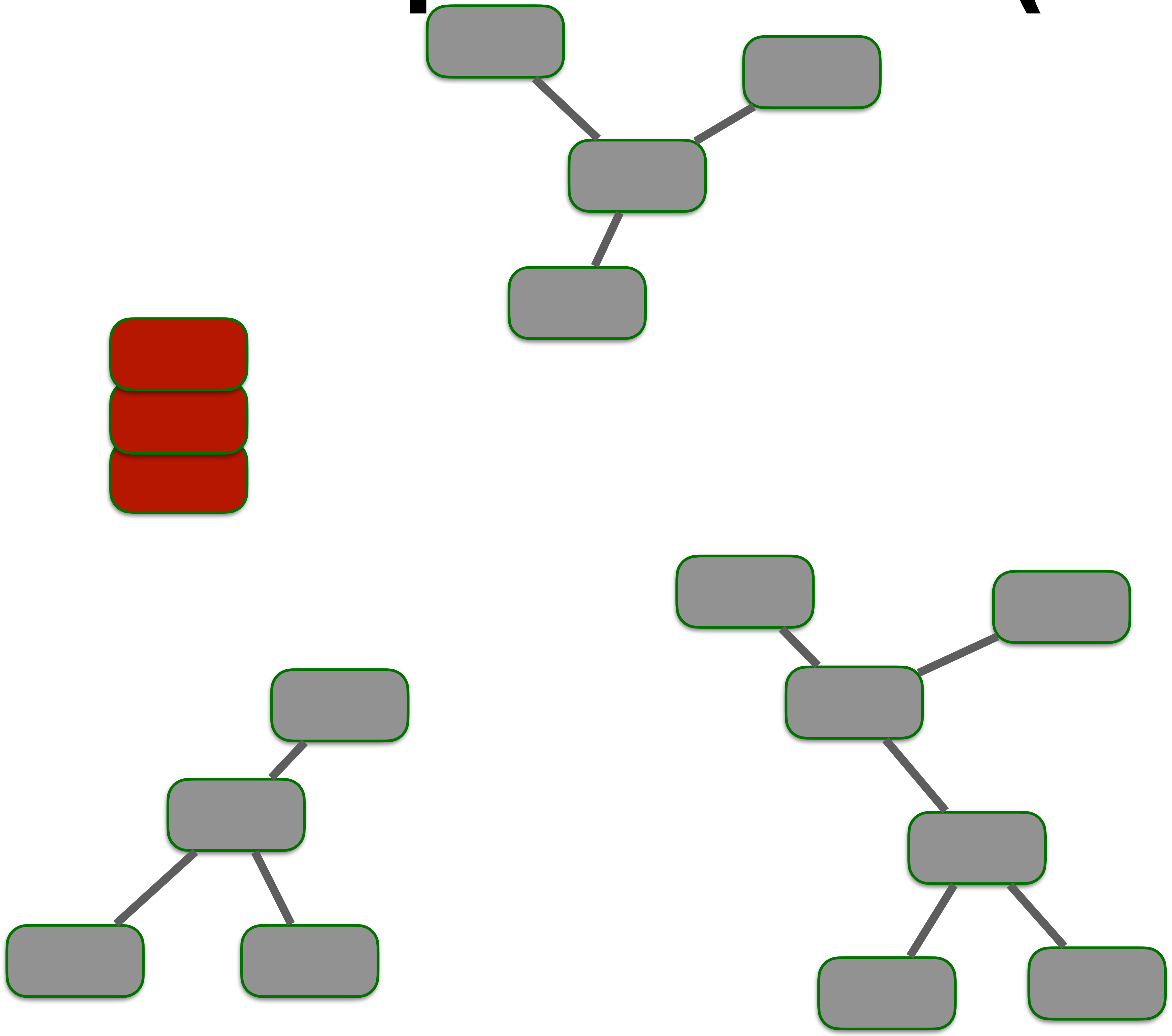
We start with a bounded treewidth BN obtained by a heuristic method

SAT-Based Local Improvement (SLIM)

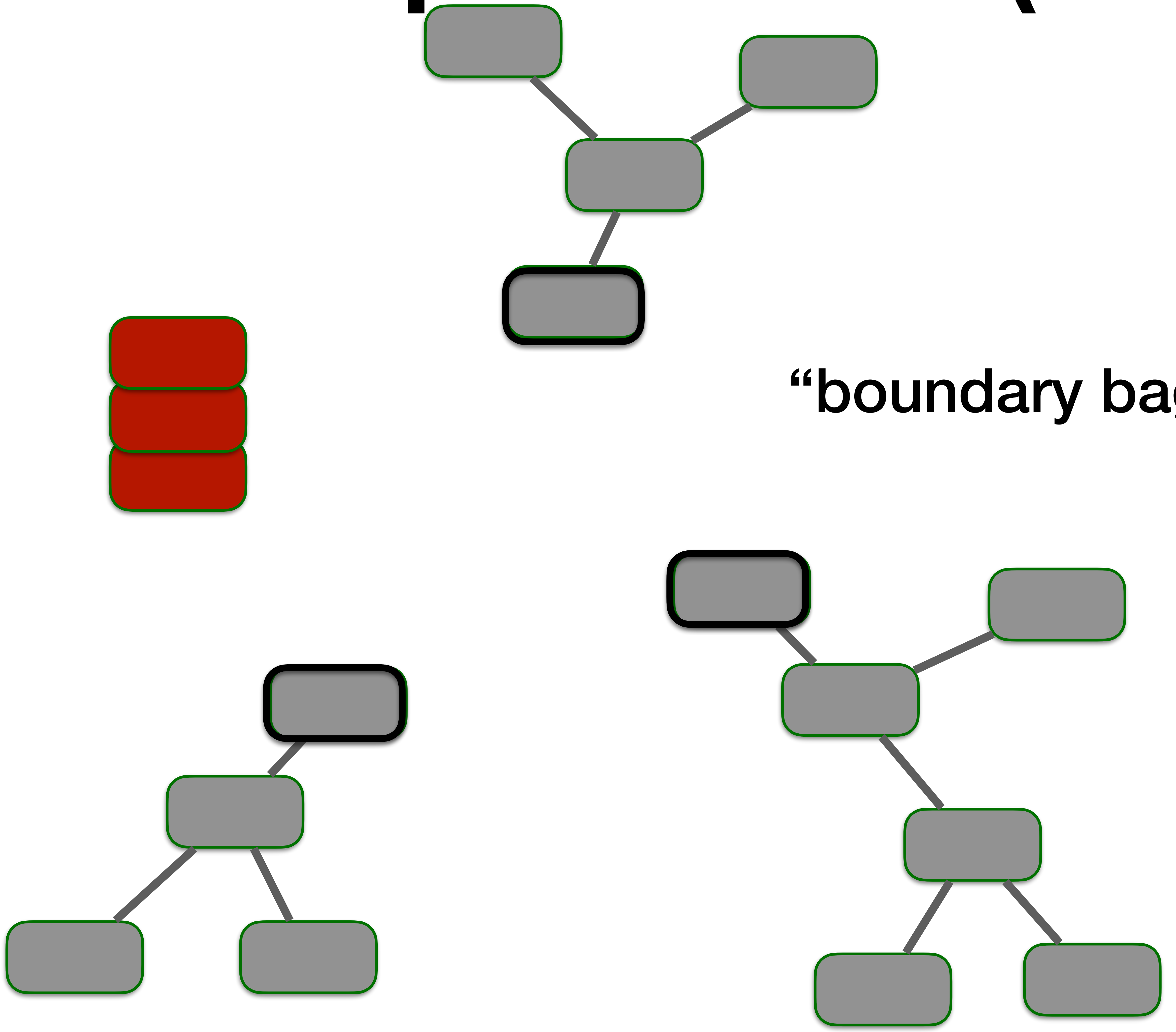


select a subtree of the tree decomposition, small enough so that the number variables in the subtree stays within a given **budget**

SAT-Based Local Improvement (SLIM)

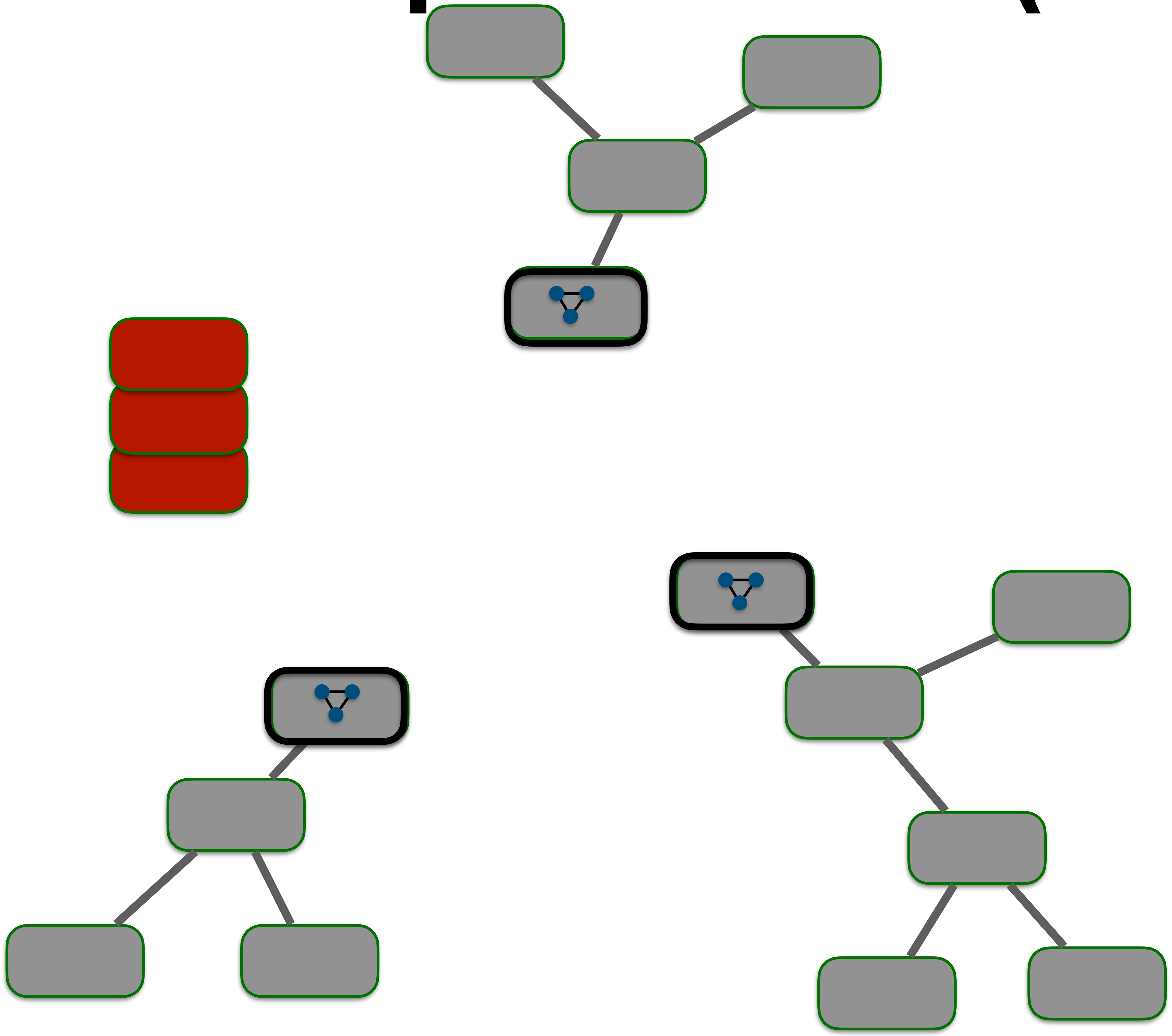


SAT-Based Local Improvement (SLIM)

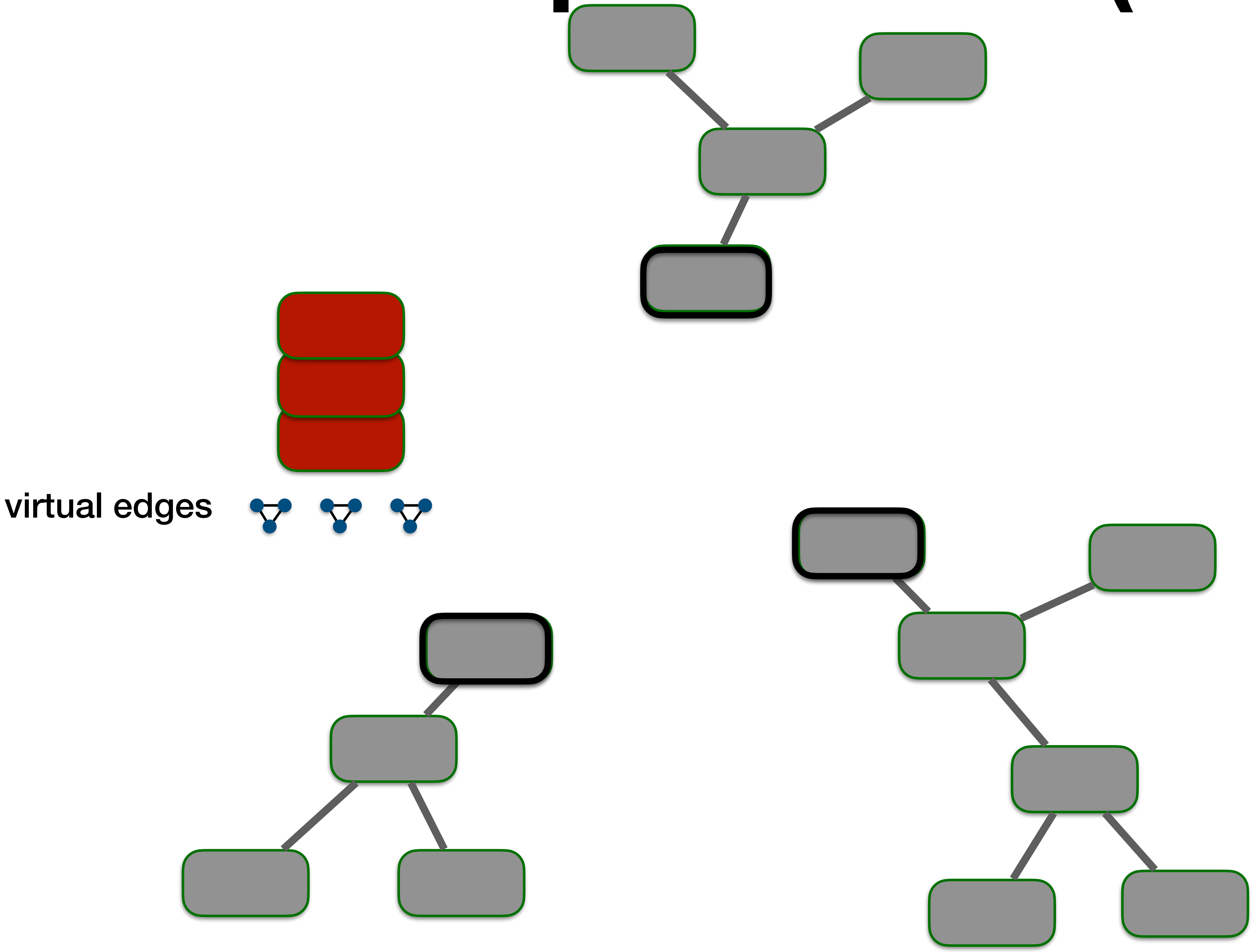


“boundary bags”

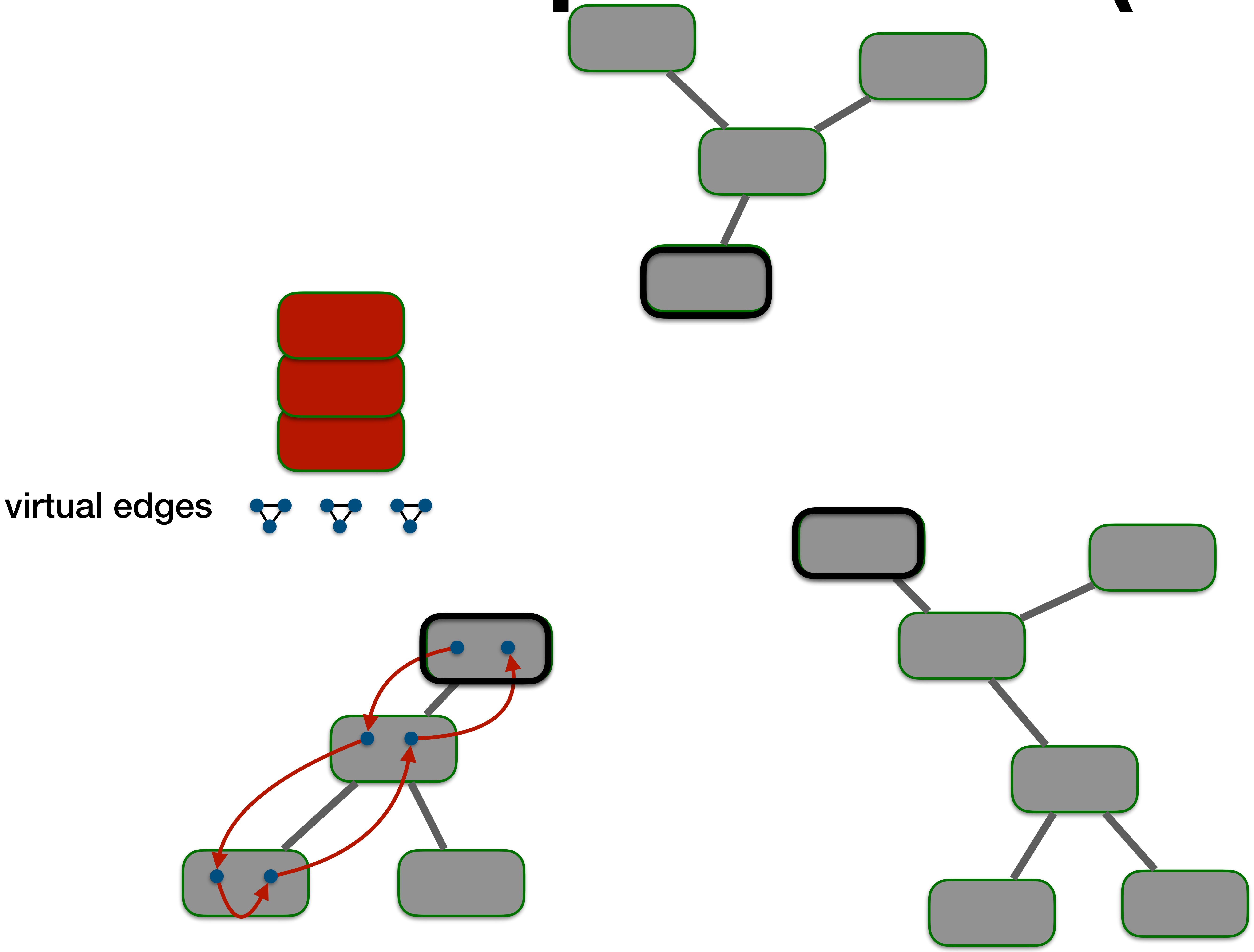
SAT-Based Local Improvement (SLIM)



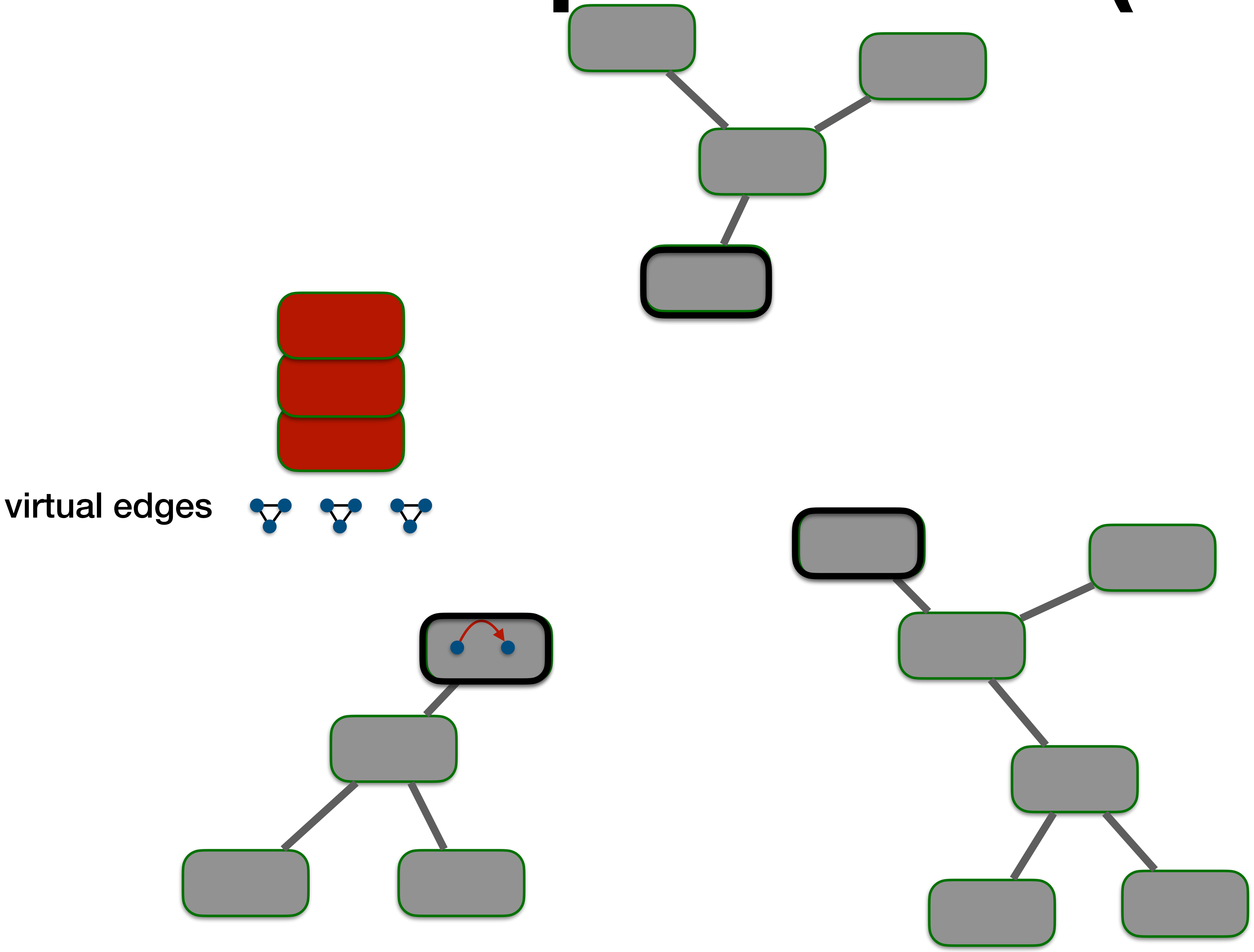
SAT-Based Local Improvement (SLIM)



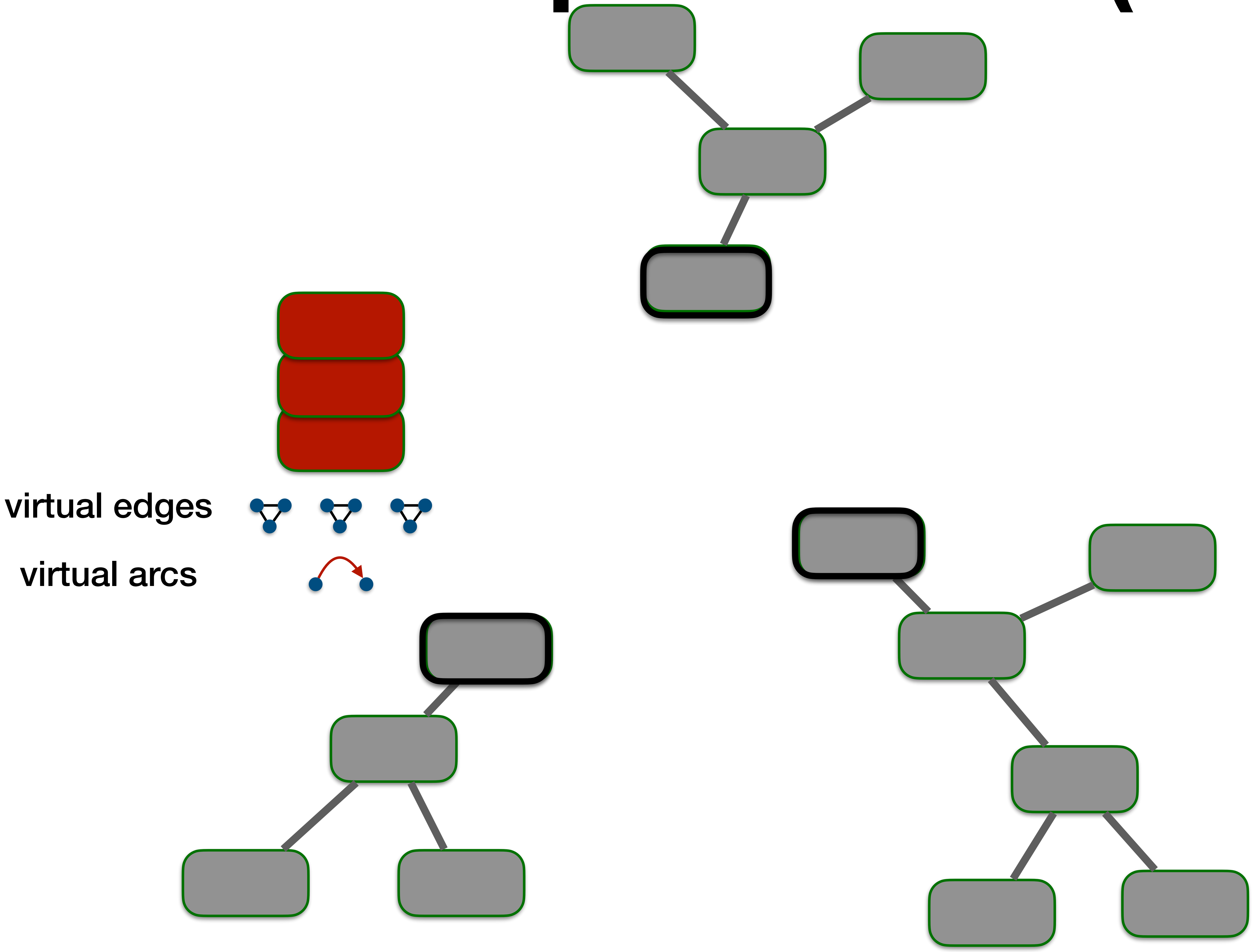
SAT-Based Local Improvement (SLIM)



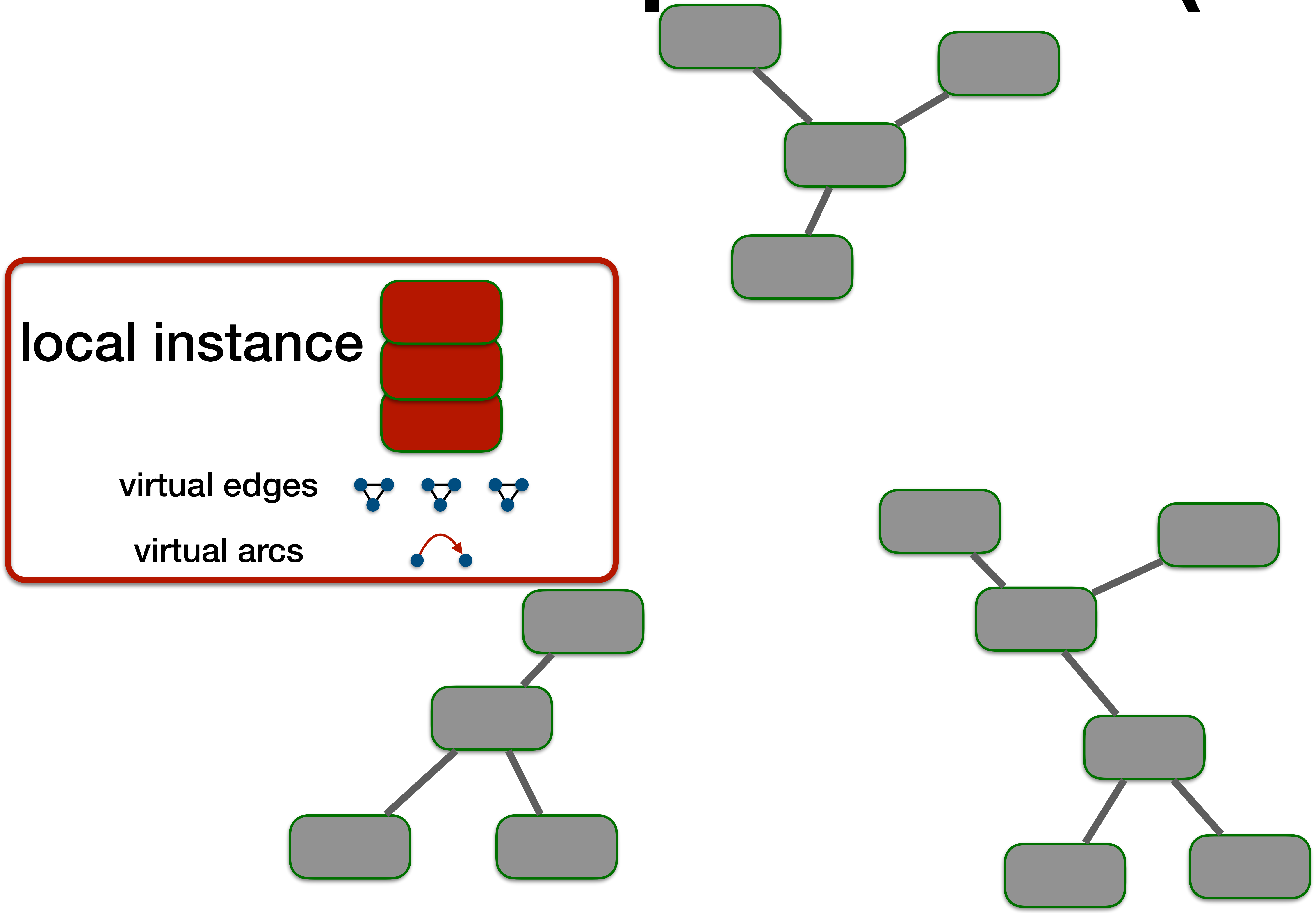
SAT-Based Local Improvement (SLIM)



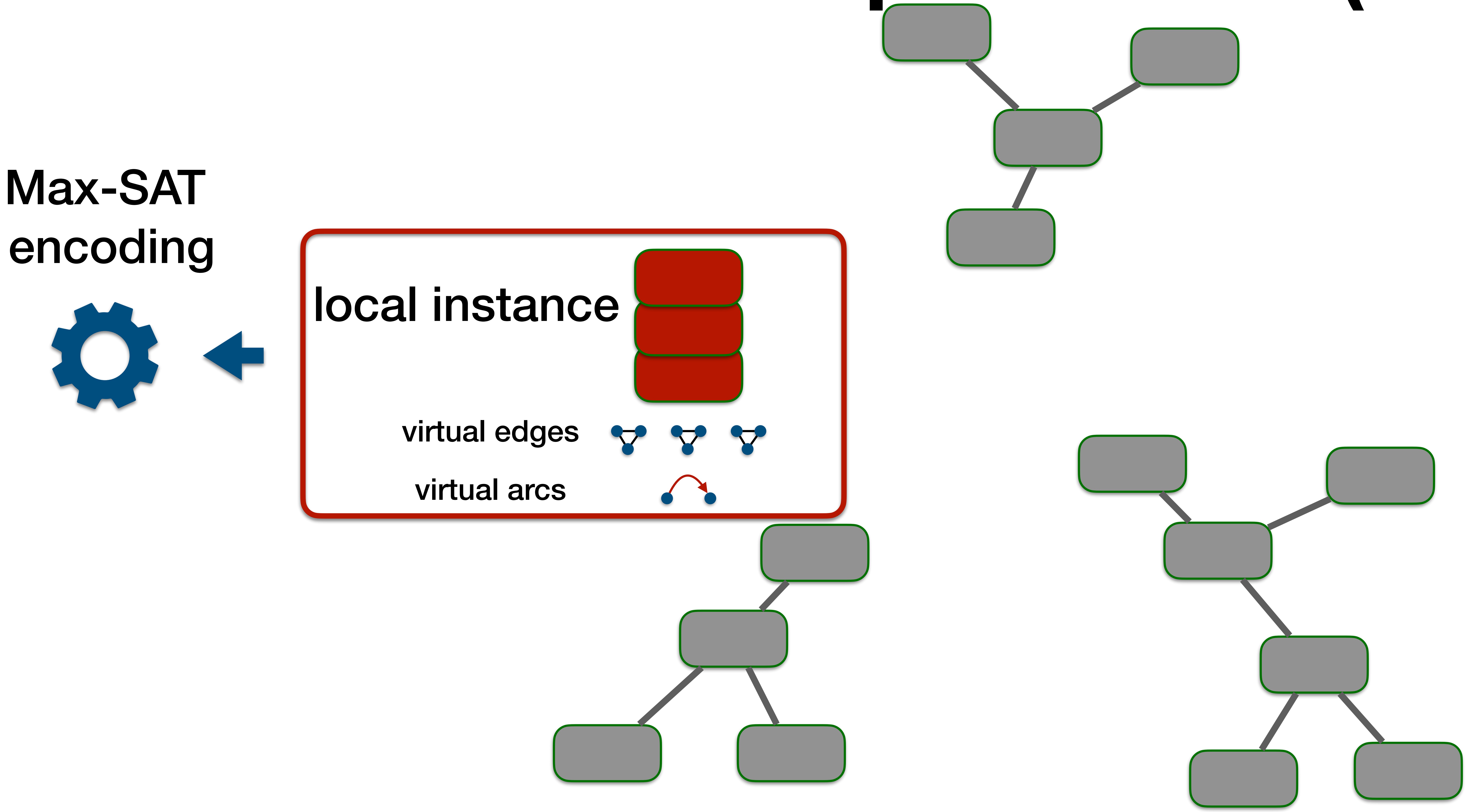
SAT-Based Local Improvement (SLIM)



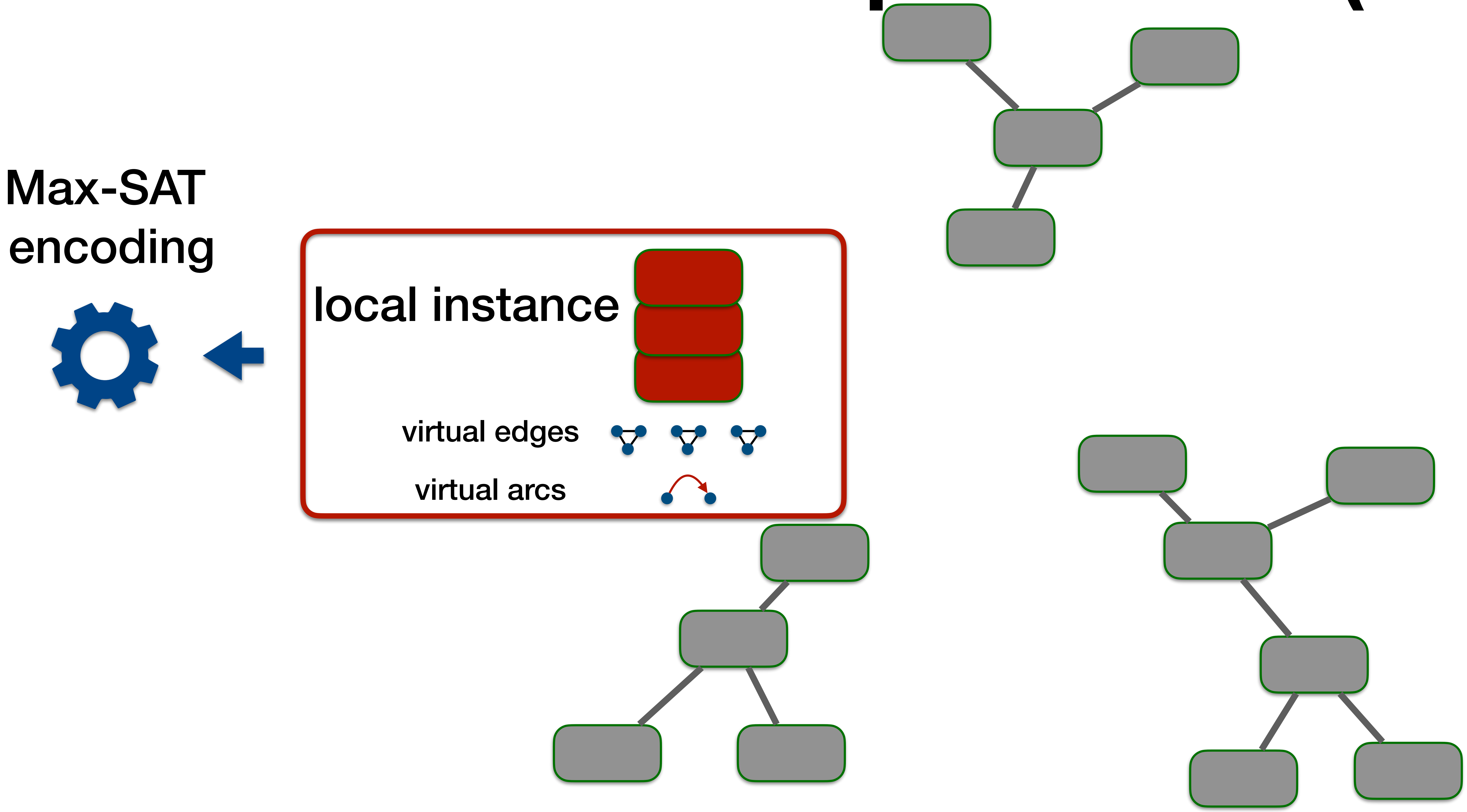
SAT-Based Local Improvement (SLIM)



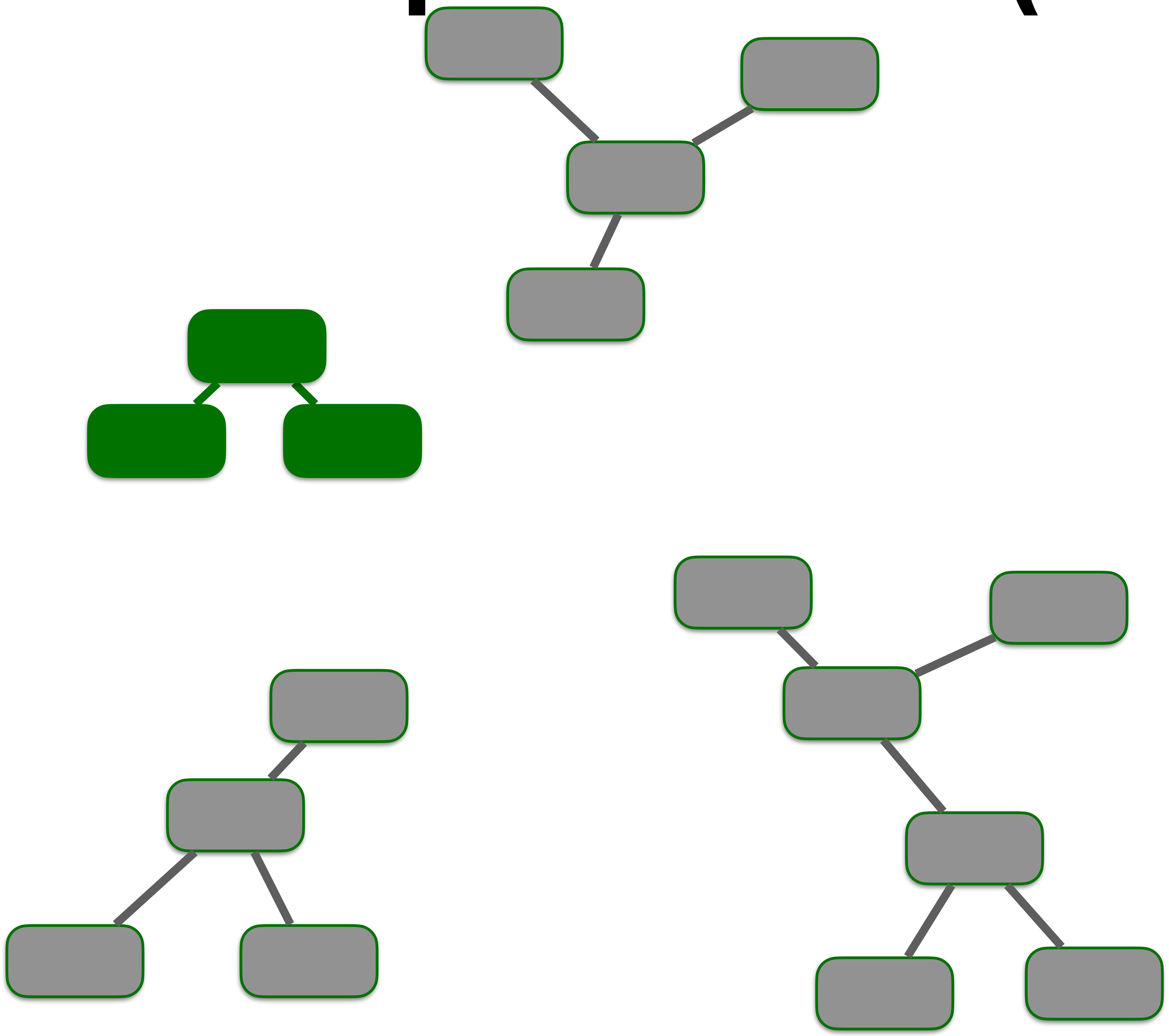
SAT-Based Local Improvement (SLIM)



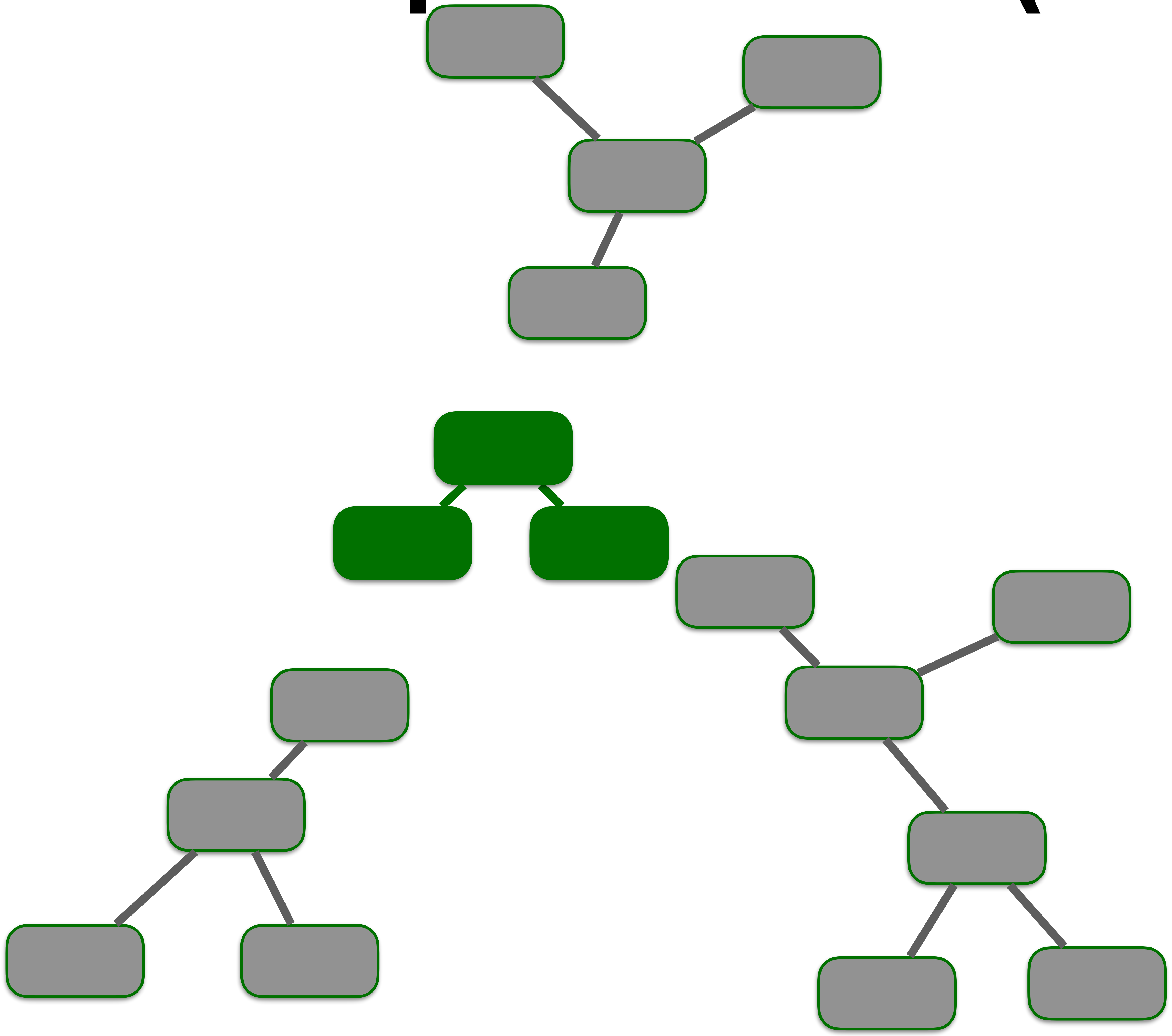
SAT-Based Local Improvement (SLIM)



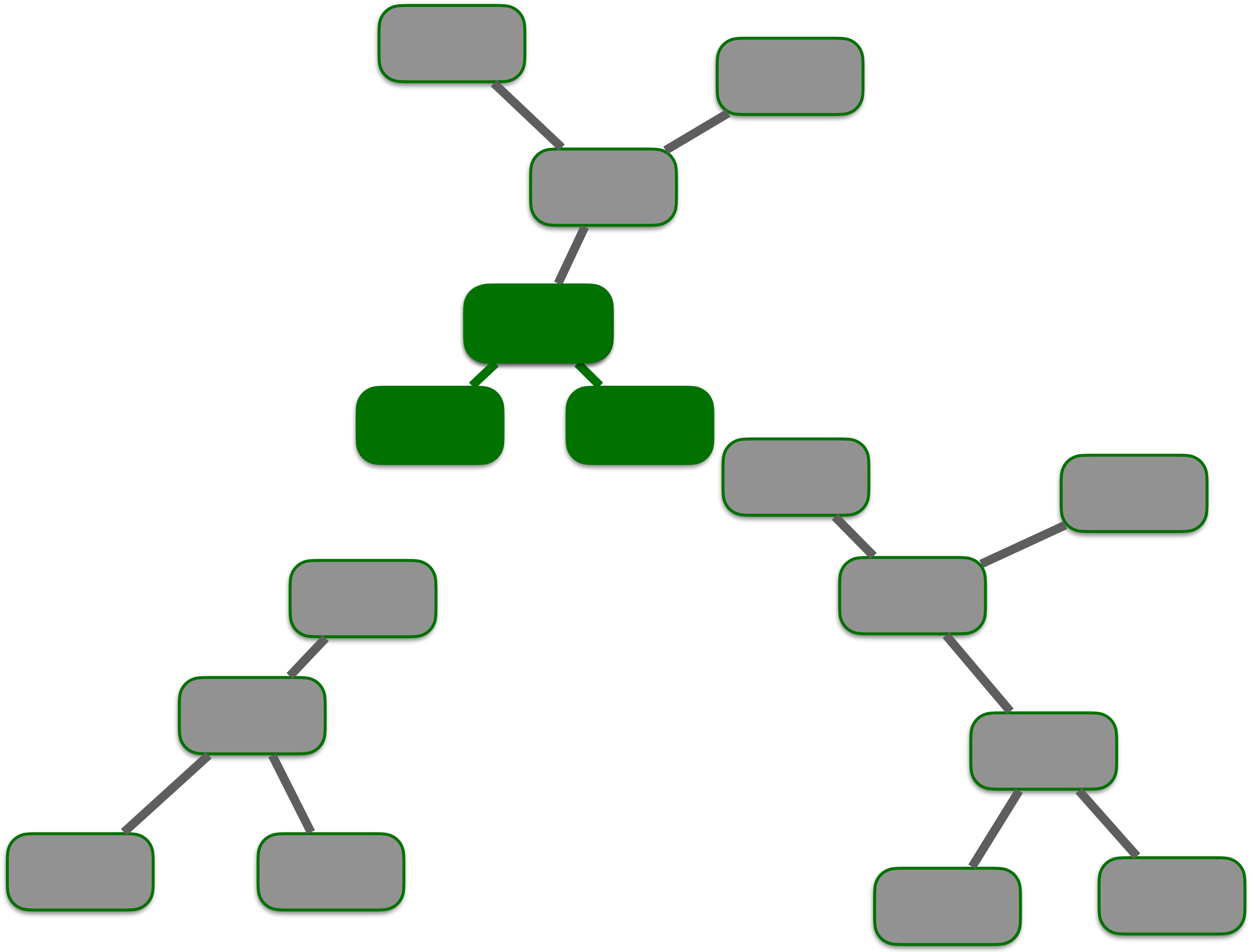
SAT-Based Local Improvement (SLIM)



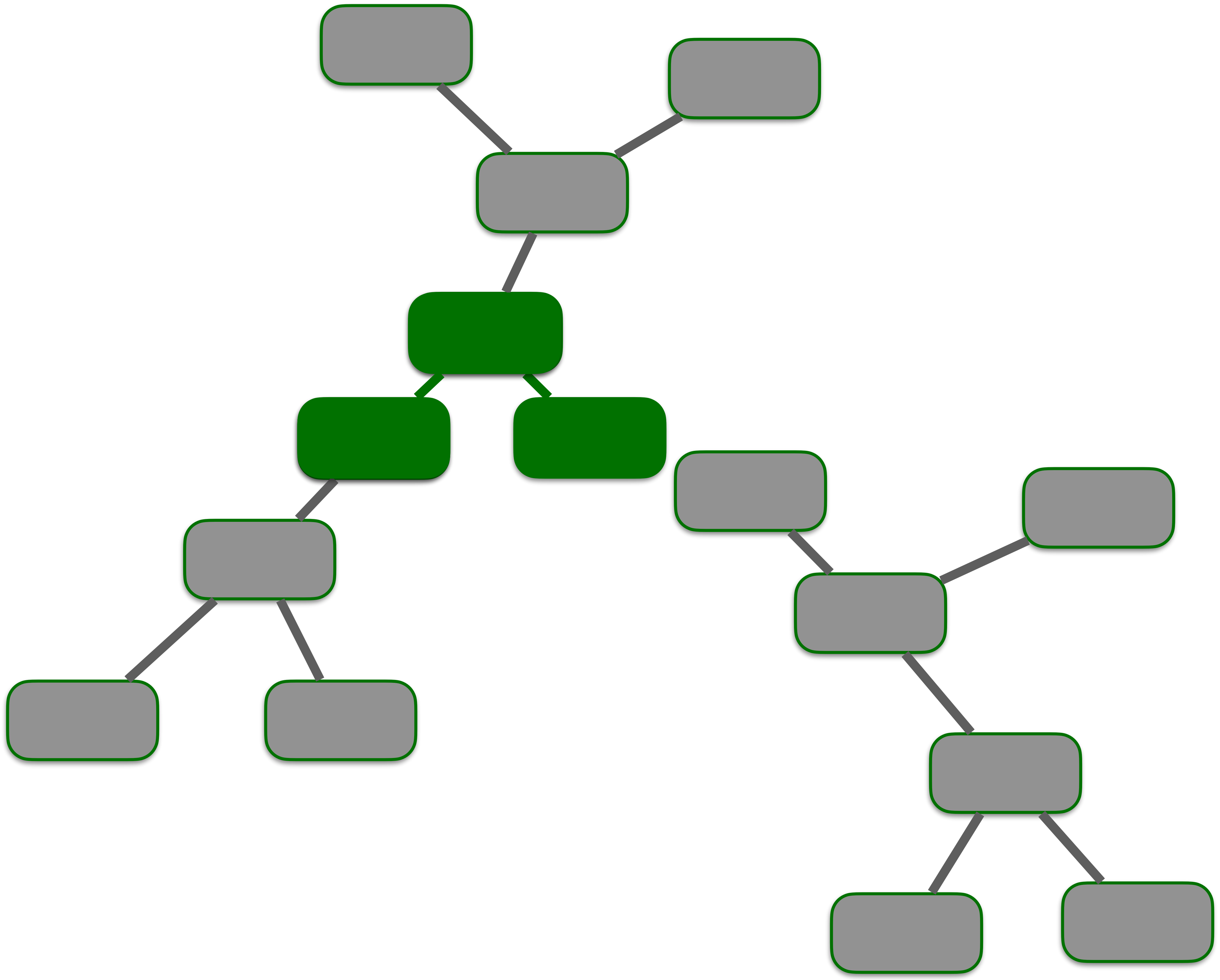
SAT-Based Local Improvement (SLIM)



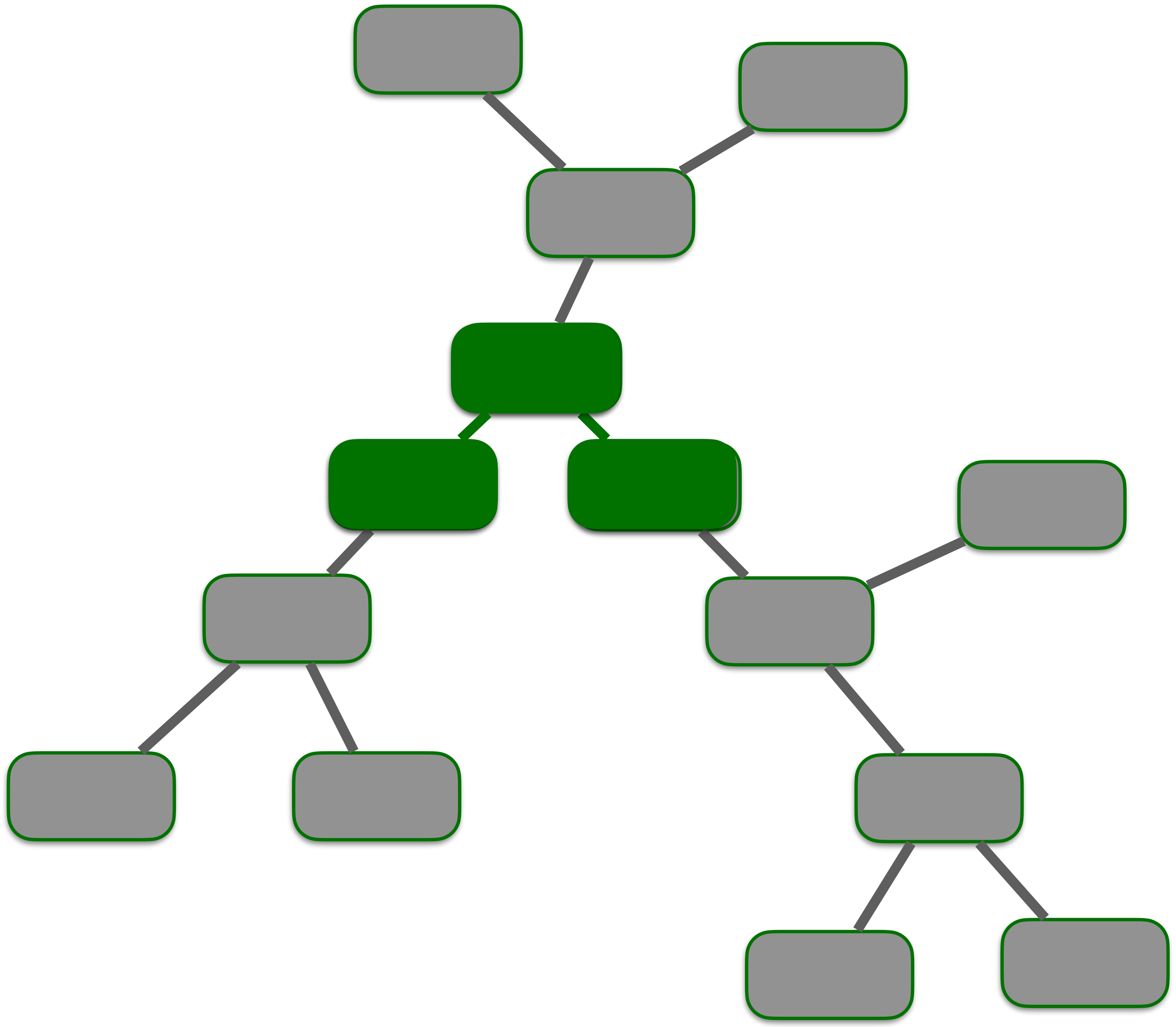
SAT-Based Local Improvement (SLIM)



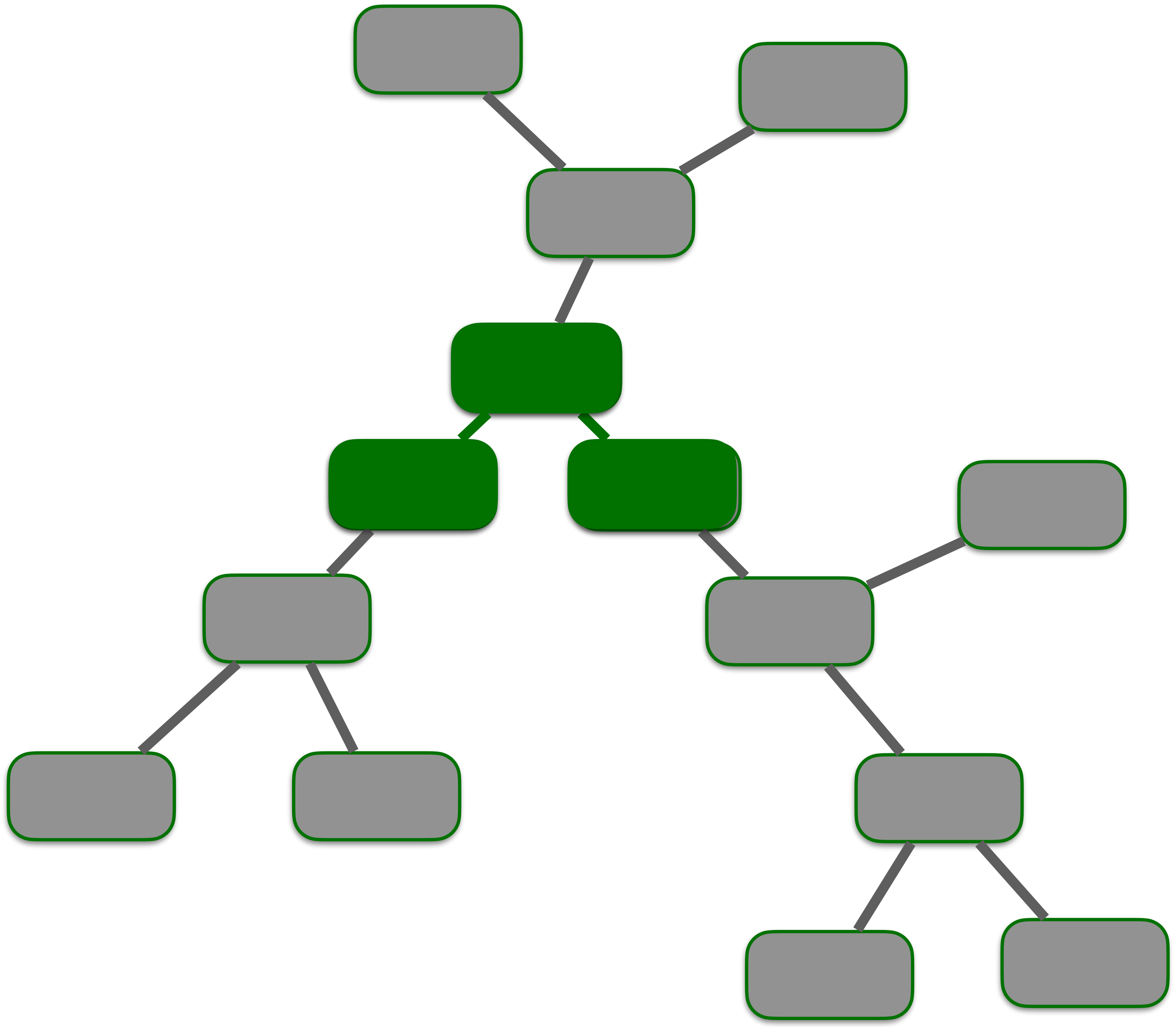
SAT-Based Local Improvement (SLIM)



SAT-Based Local Improvement (SLIM)



SAT-Based Local Improvement (SLIM)



we repeat this process
for other selected
subtrees, until no
improvement is possible
or a global timeout is
reached

Experimental Setup

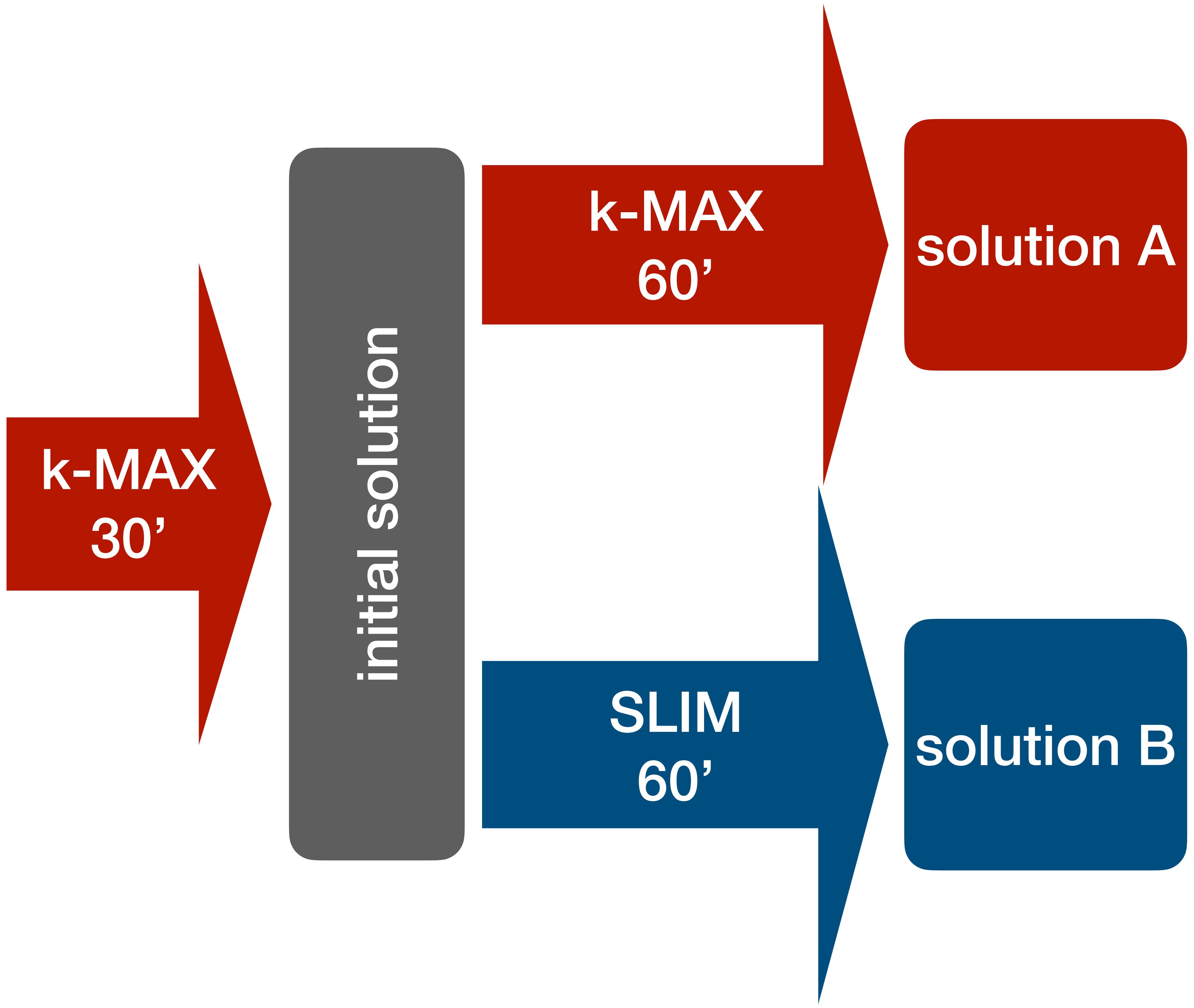
- **Instances:** up to 4000 variables
- **k-MAX algorithm [Scanagatta et al. 2018]**
  - anytime algorithm, we take the initial solution after 30 minutes
- **Treewidth bounds**: we use tw=2, tw=5, and tw=8 as in
- **MAX-SAT solver:** UWrMaxSat [Piotrow 2019] because of its **anytime behaviour,** good performance in 2019 MaxSAT evaluation. We don't run it until termination
- **Local timeout:** 2 seconds
- **Budget:** 10 tree nodes

SLIM Turbocharging k-MAX

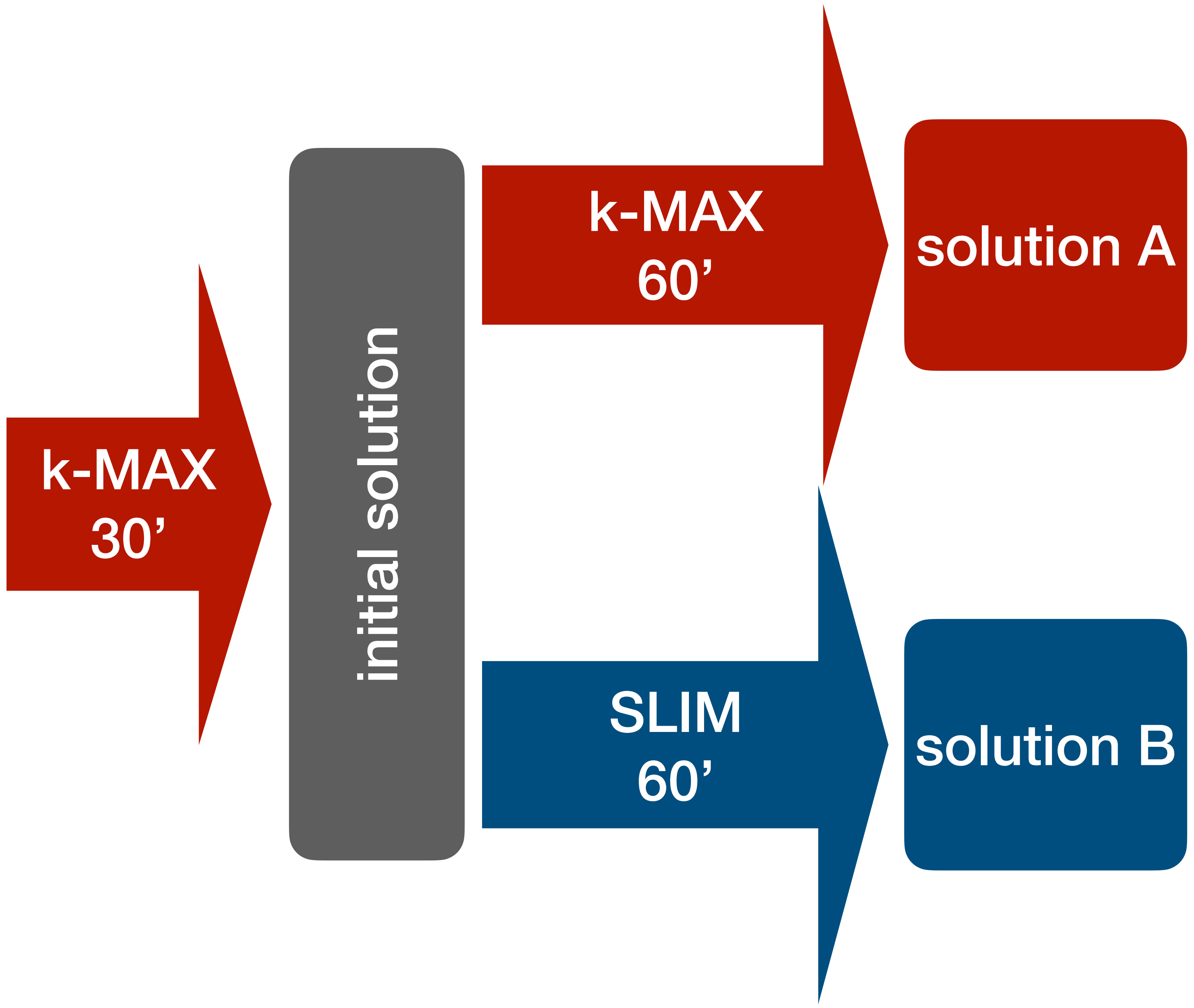
k-MAX
30’

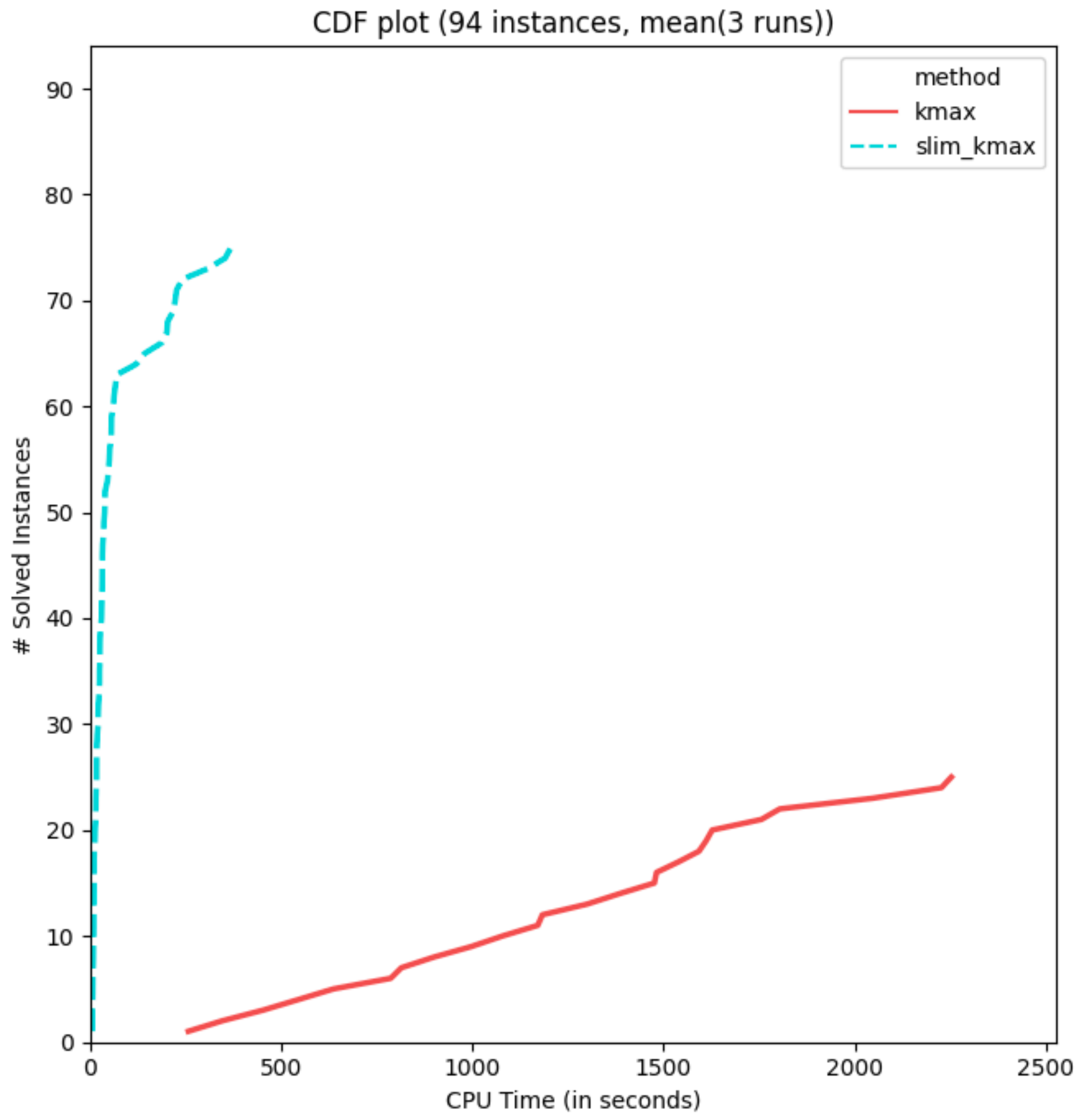
SLIM Turbocharging k-MAX

SLIM Turbocharging k-MAX

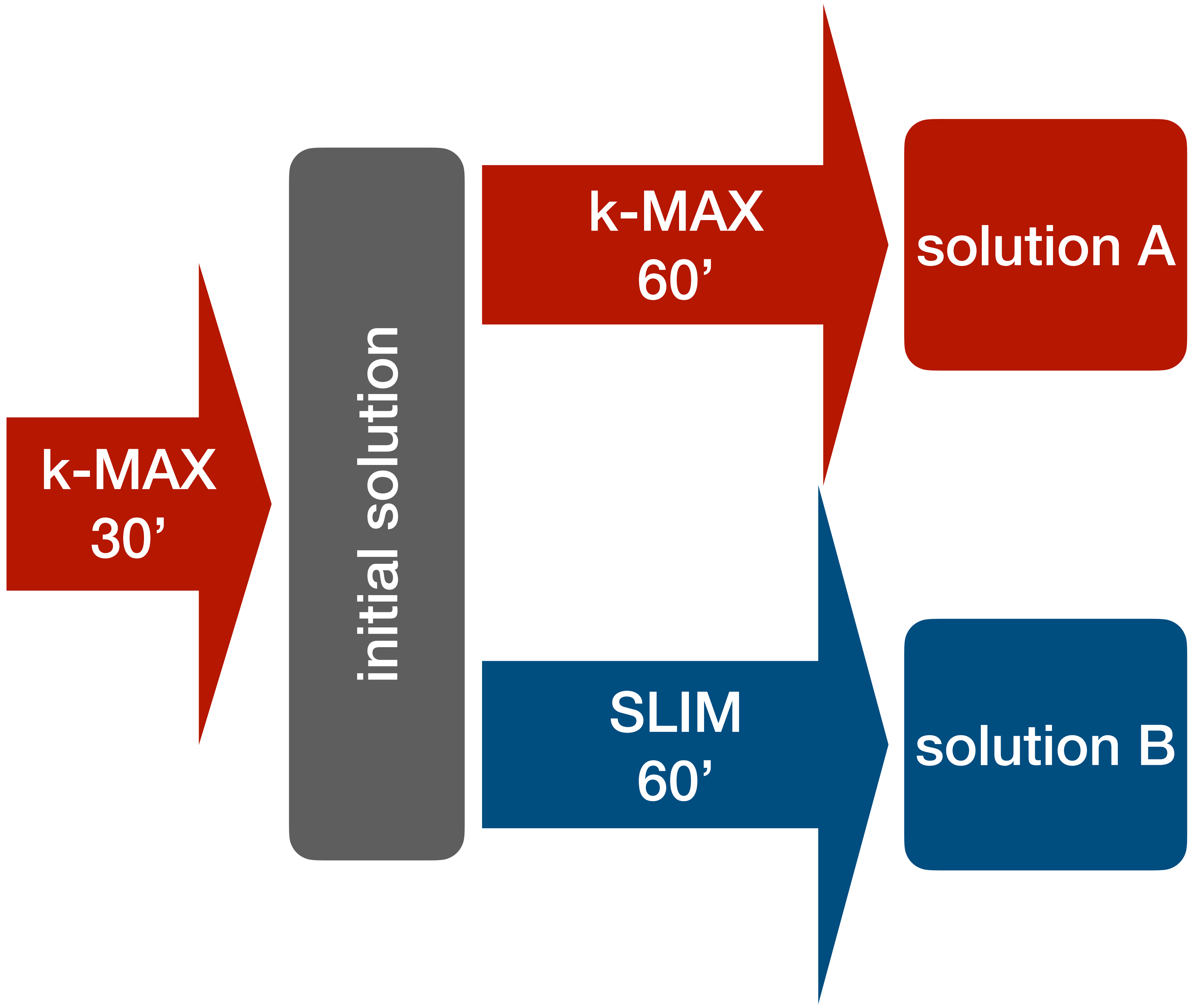


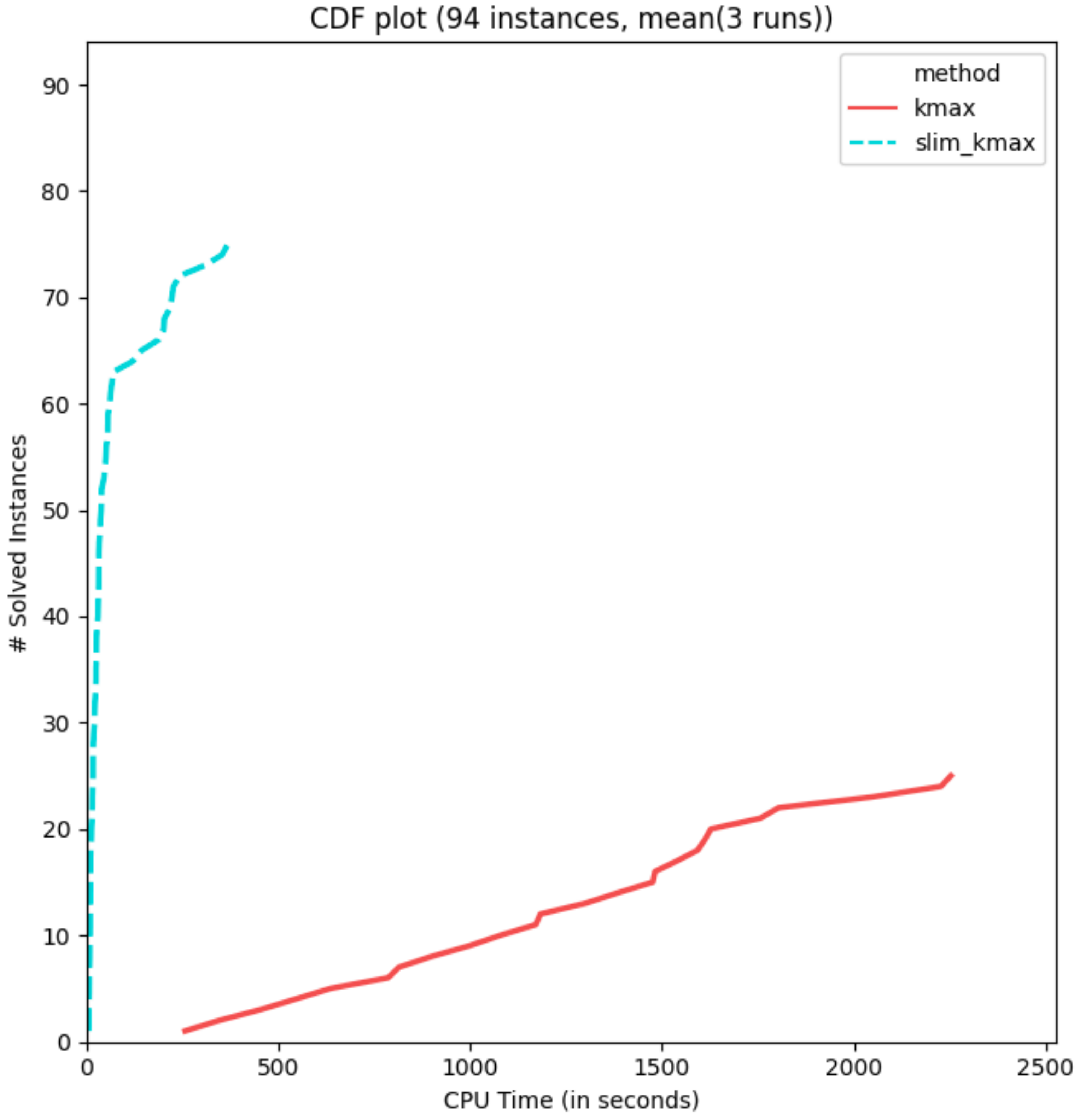
SLIM Turbocharging k-MAX





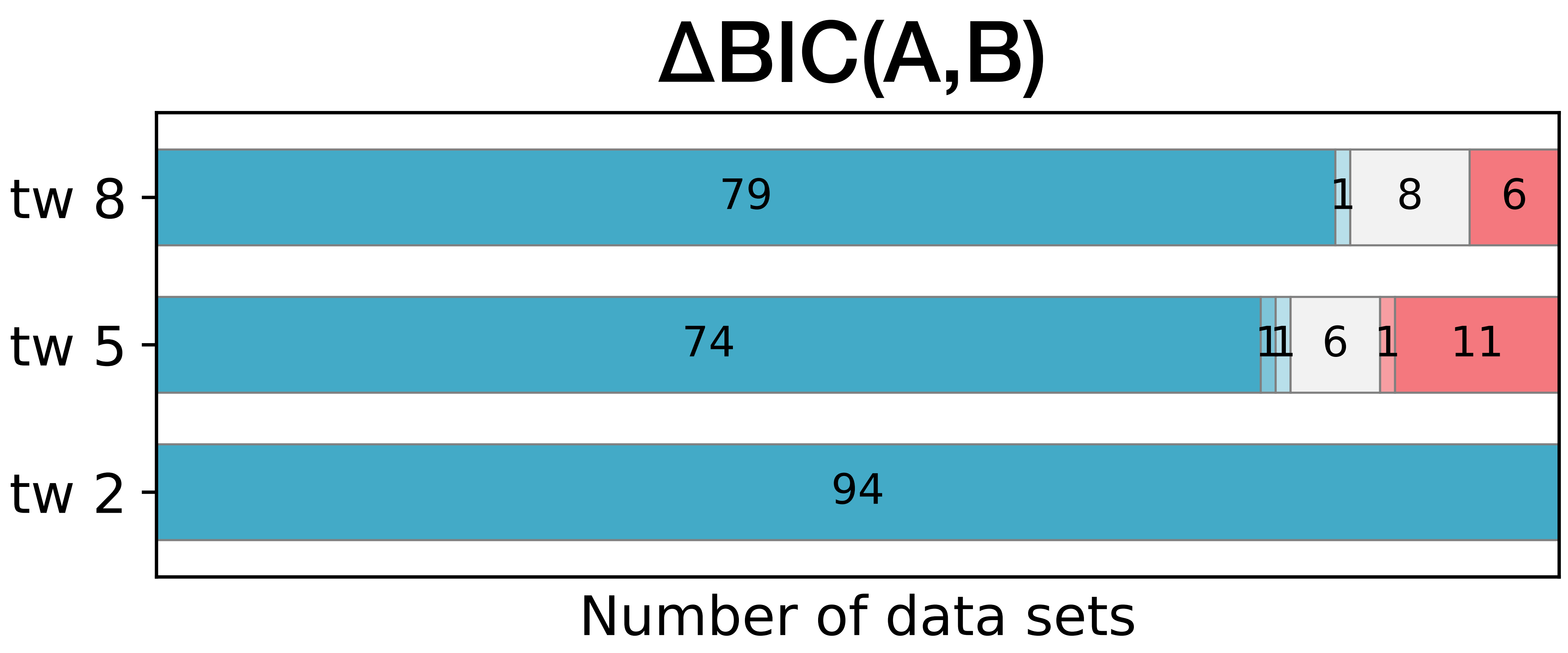
SLIM Turbocharging k-MAX





- k-MAX+SLIM outperforms kMAX on a significant number of instances
- and on all instances with tw=2 bound

Side-by-Side Comparison

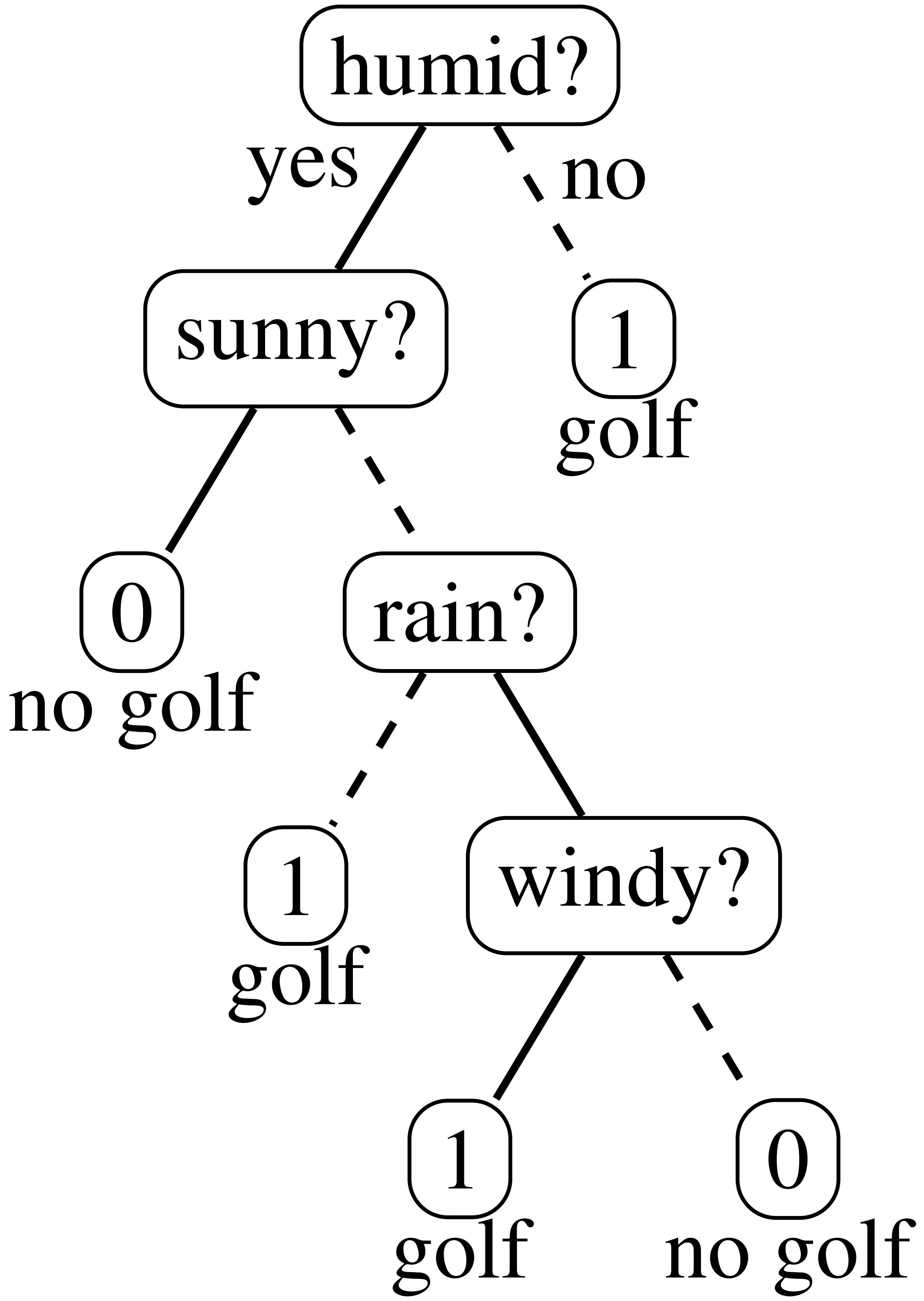


	Category	ΔBIC		Category	ΔBIC
	extremely neg.	$(-\infty, -10)$		extremely pos.	$(10, \infty)$
	strongly neg.	$(-10, -6)$		strongly pos.	$(6, 10)$
	negative	$(-6, -2)$		positive	$(2, 6)$

- The **BIC score** approximates the logarithm of the marginal likelihood of a DAG, i.e., the Bayes Factor [Raftery 1995]

Decision Tree Induction

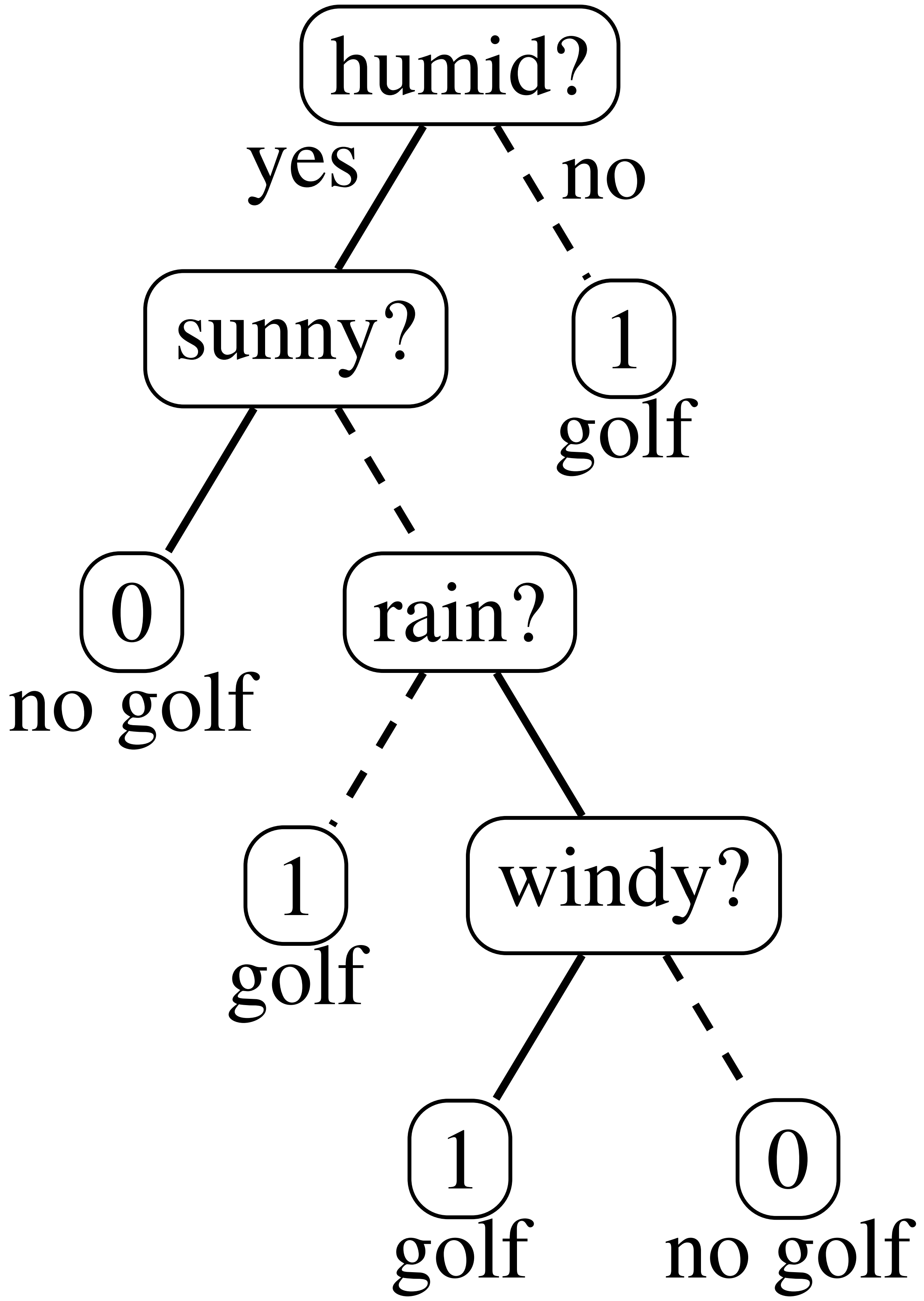
Design Trees



- Established tool for the description, classification, and generalization of data.
- Easy to interpret: the path from root to leaf provides an explanation. Important in the context of explainable AI.
- The depth of the tree determines the maximum length of explanations.
- Smaller decision trees usually generalize better.

Design Trees

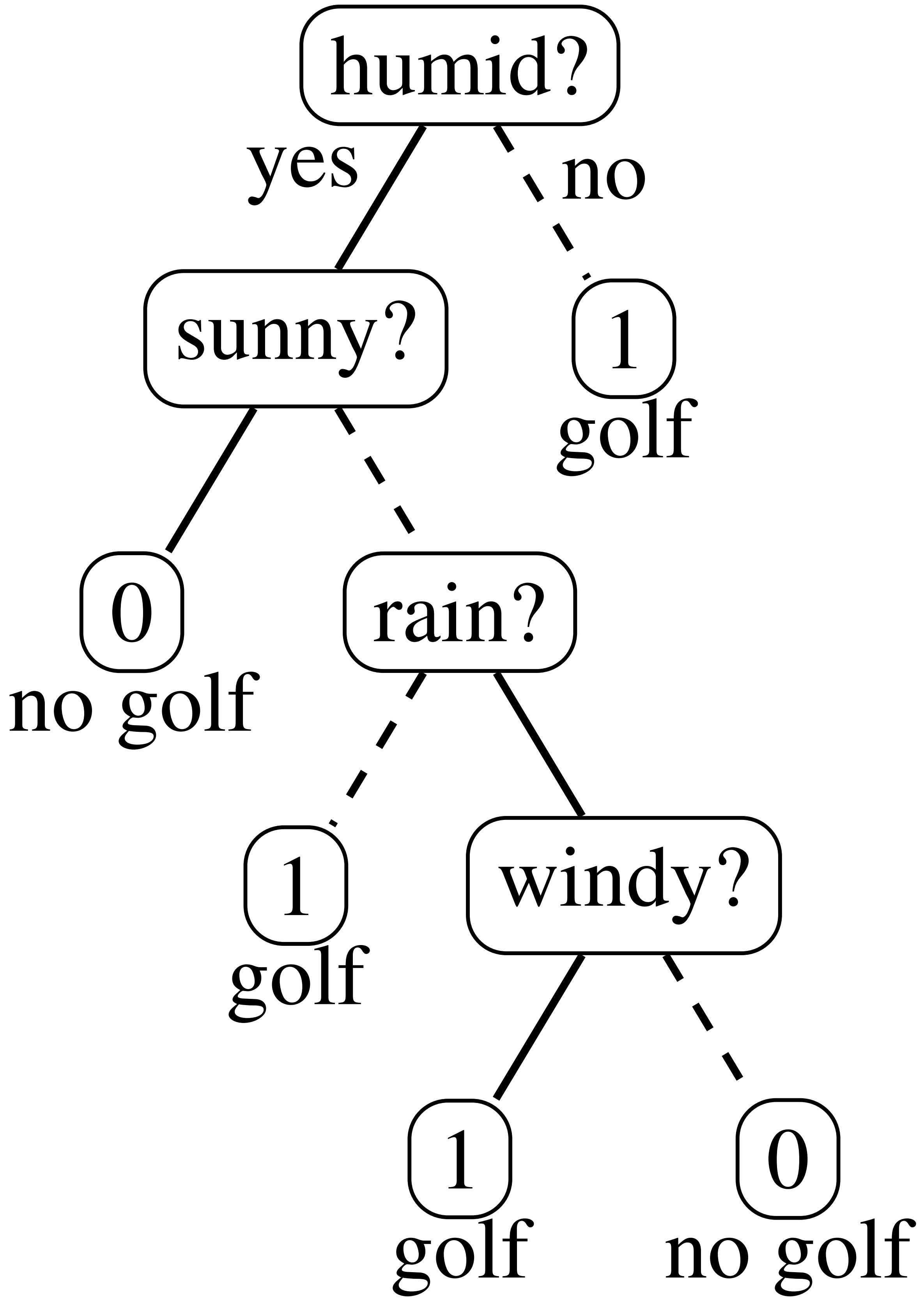
sample	sunny	rain	overcast	temp:mild	temp:hot	temp:cool	humid	windy	play golf
e_1	0	1	0	0	0	1	0	1	1
e_2	0	1	0	1	0	0	0	1	1
e_3	1	0	0	0	1	0	1	1	0
e_4	0	0	1	0	0	1	0	0	1
e_5	1	0	0	1	0	0	0	0	1
e_6	0	1	0	1	0	0	1	0	0
e_7	0	1	0	1	0	0	1	1	1
e_8	0	0	1	0	1	0	1	1	1
e_9	1	0	0	1	0	0	1	1	0
e_{10}	1	0	0	0	0	1	0	1	1
e_{11}	0	0	1	1	0	0	1	0	1
e_{12}	1	0	0	0	1	0	1	0	0

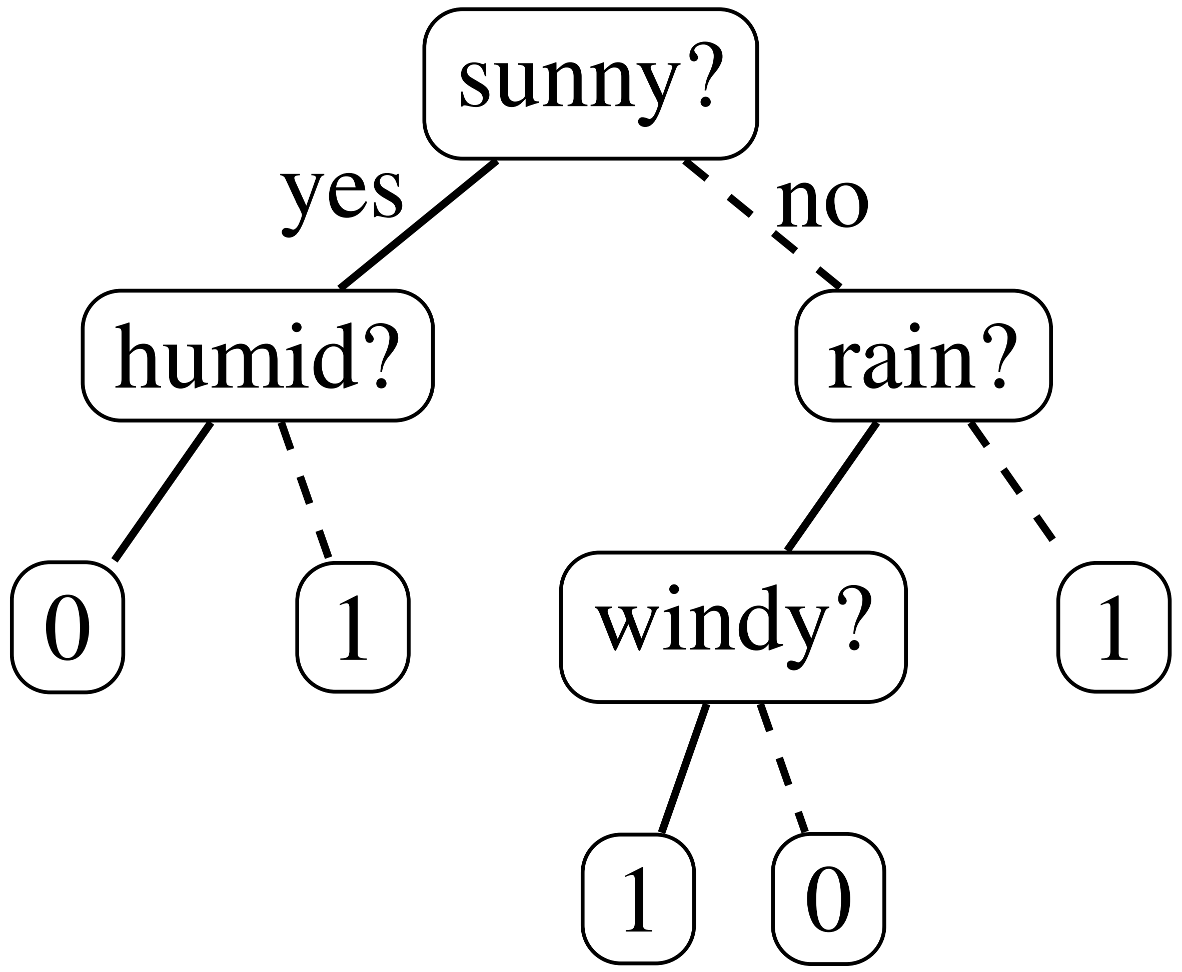


- Established tool for the description, classification, and generalization of data.
- Easy to interpret: the path from root to leaf provides an explanation. Important in the context of explainable AI.
- The depth of the tree determines the maximum length of explanations.
- Smaller decision trees usually generalize better.

Design Trees

sample	sunny	rain	overcast	temp:mild	temp:hot	temp:cool	humid	windy	play golf
e_1	0	1	0	0	0	1	0	1	1
e_2	0	1	0	1	0	0	0	1	1
e_3	1	0	0	0	1	0	1	1	0
e_4	0	0	1	0	0	1	0	0	1
e_5	1	0	0	1	0	0	0	0	1
e_6	0	1	0	1	0	0	1	0	0
e_7	0	1	0	1	0	0	1	1	1
e_8	0	0	1	0	1	0	1	1	1
e_9	1	0	0	1	0	0	1	1	0
e_{10}	1	0	0	0	0	1	0	1	1
e_{11}	0	0	1	1	0	0	1	0	1
e_{12}	1	0	0	0	1	0	1	0	0





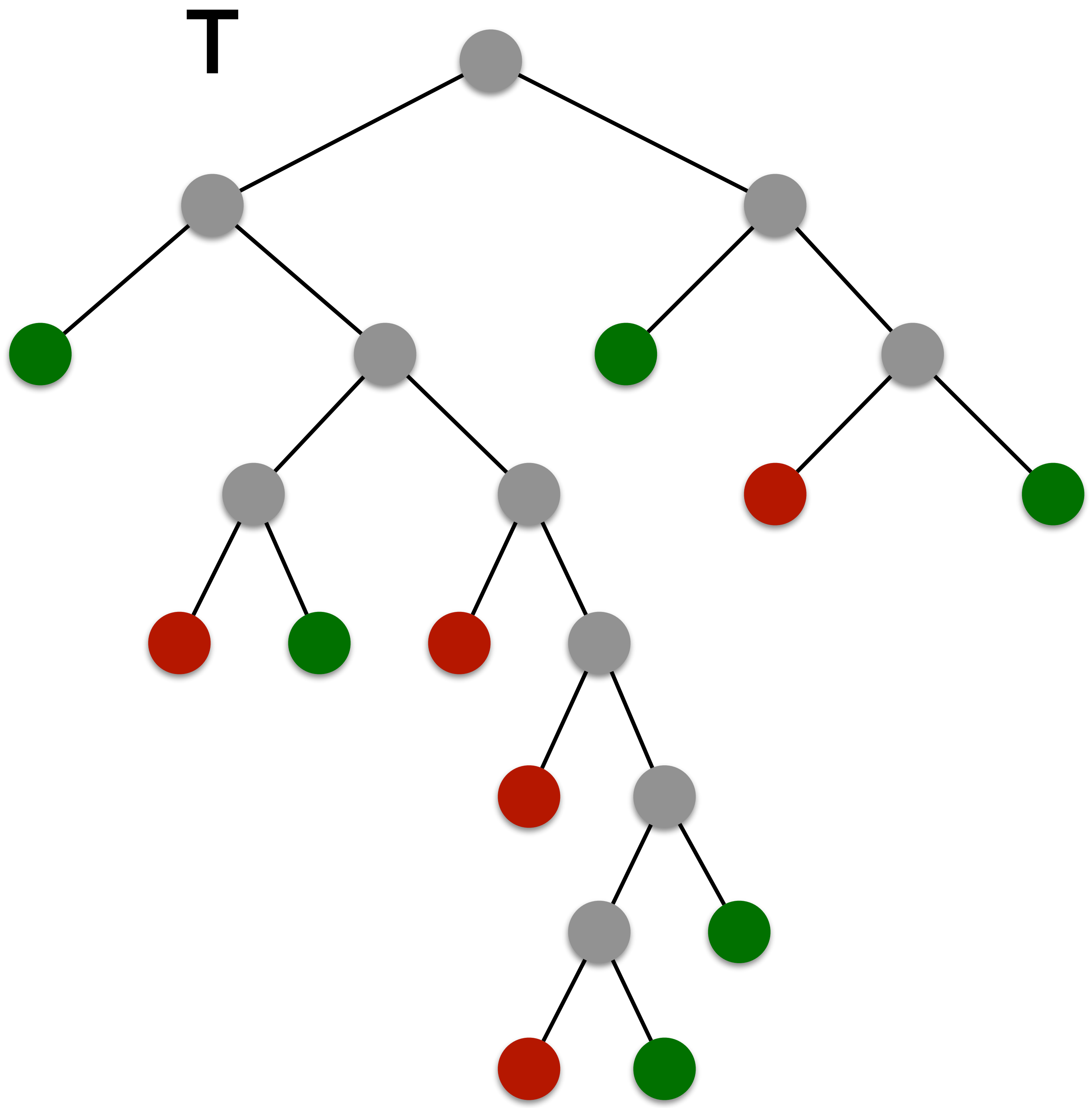
SAT encodings for DT learning

- First SAT encoding for decision trees did not scale to even small instances [Bessiere, Hebrard, O'Sullivan CP'09]
- First encoding that scaled to small instances [Narodytska, Ignatiev, Pereira, Marques-Silva IJCAI'18]
- Depth-based encoding that scales to larger instances [Avellaneda AAAI'20].
- New partition-based encoding and SLIM approach [Schidler, Szeider AAAI'21]
- Survey paper [Ignatiev, Marques-Silva, Narodytska, Stuckey IJCAI'21]

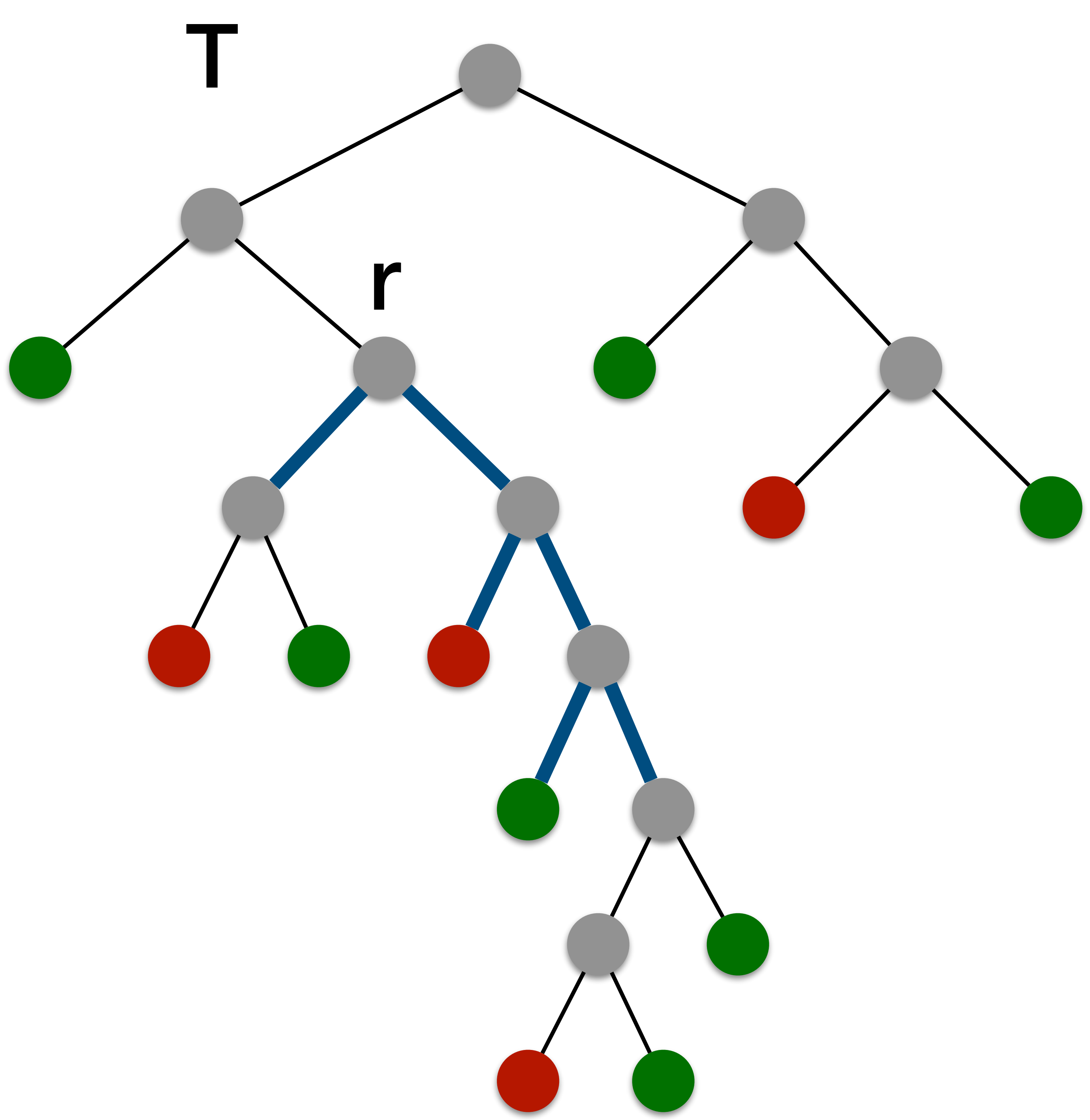
DT-SLIM

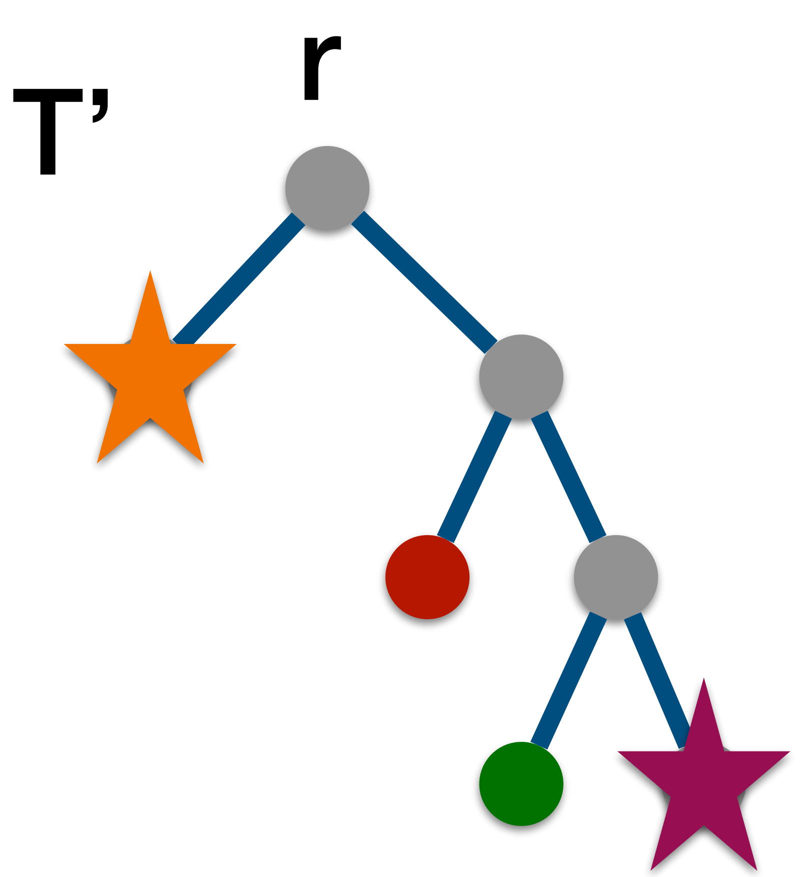
- Can be used to combine any DT heuristics (like C4.5, ITI) with an exact method (like SAT encoding)
- Requirement for the SAT encoding is that its can deal with more than 2 classification labels
- Scales virtually to any size or depth

Improvement Step



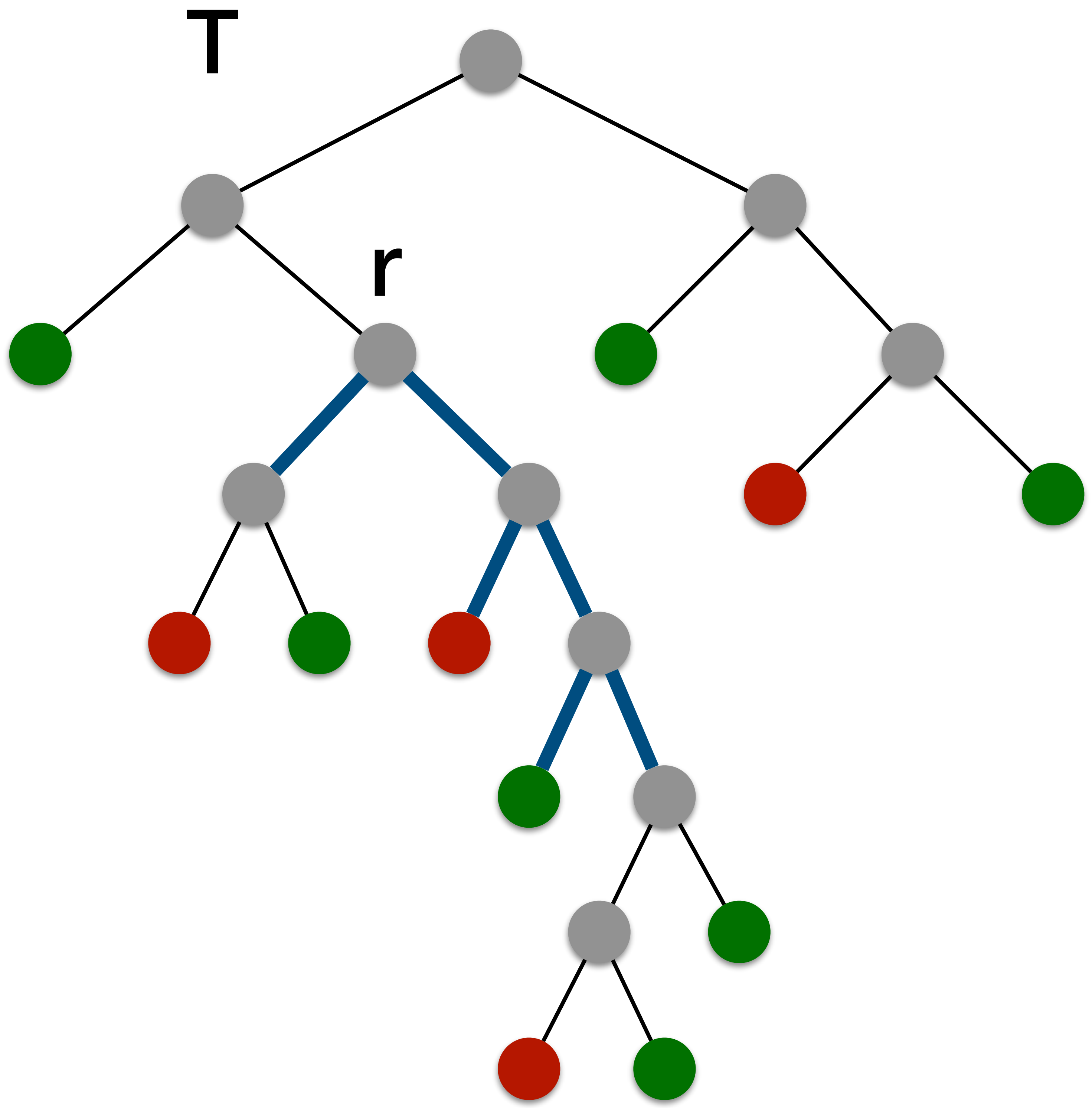
Improvement Step

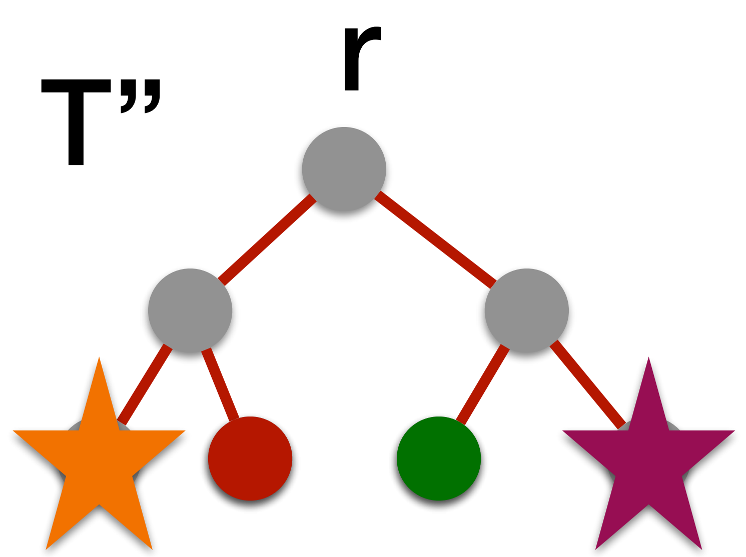




Local Instance:
all samples that end up in r,
reclassify those that end up in a
special leaf

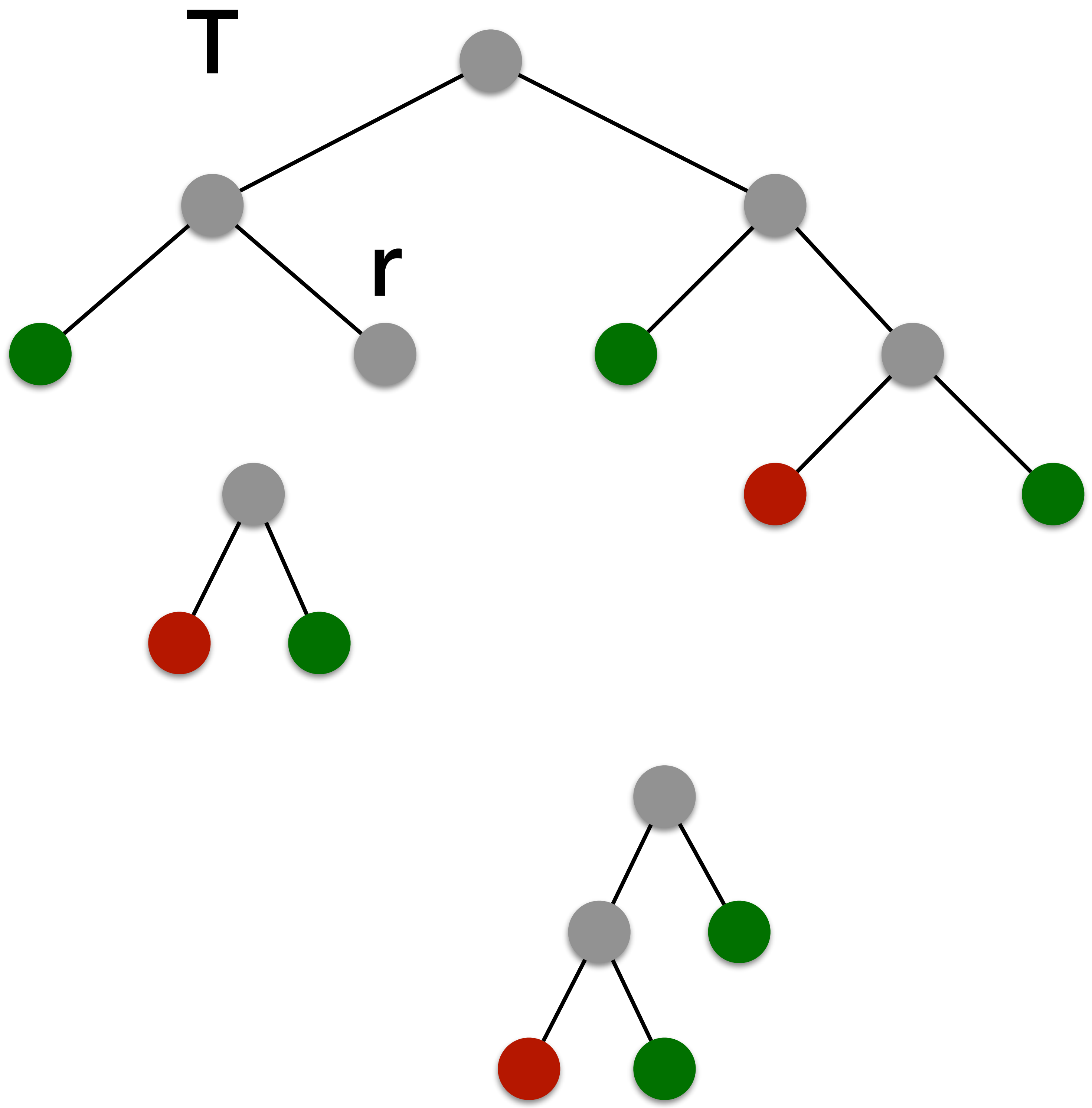
Improvement Step

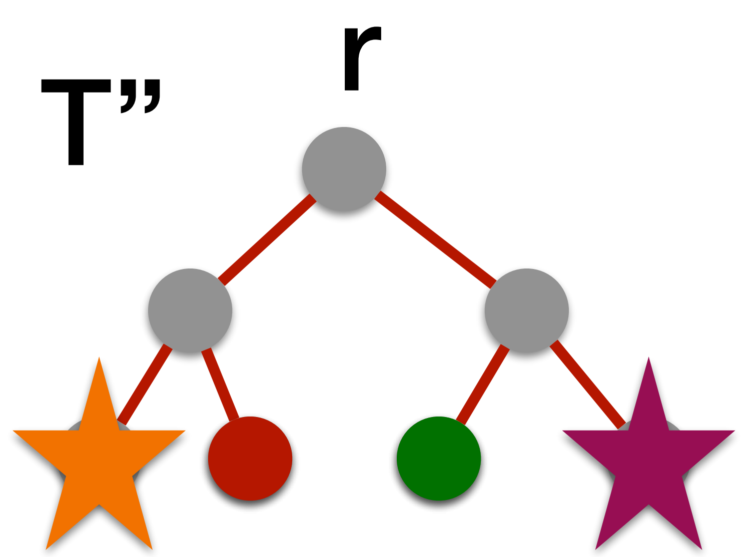




Find better tree T” for Local Instance using SAT

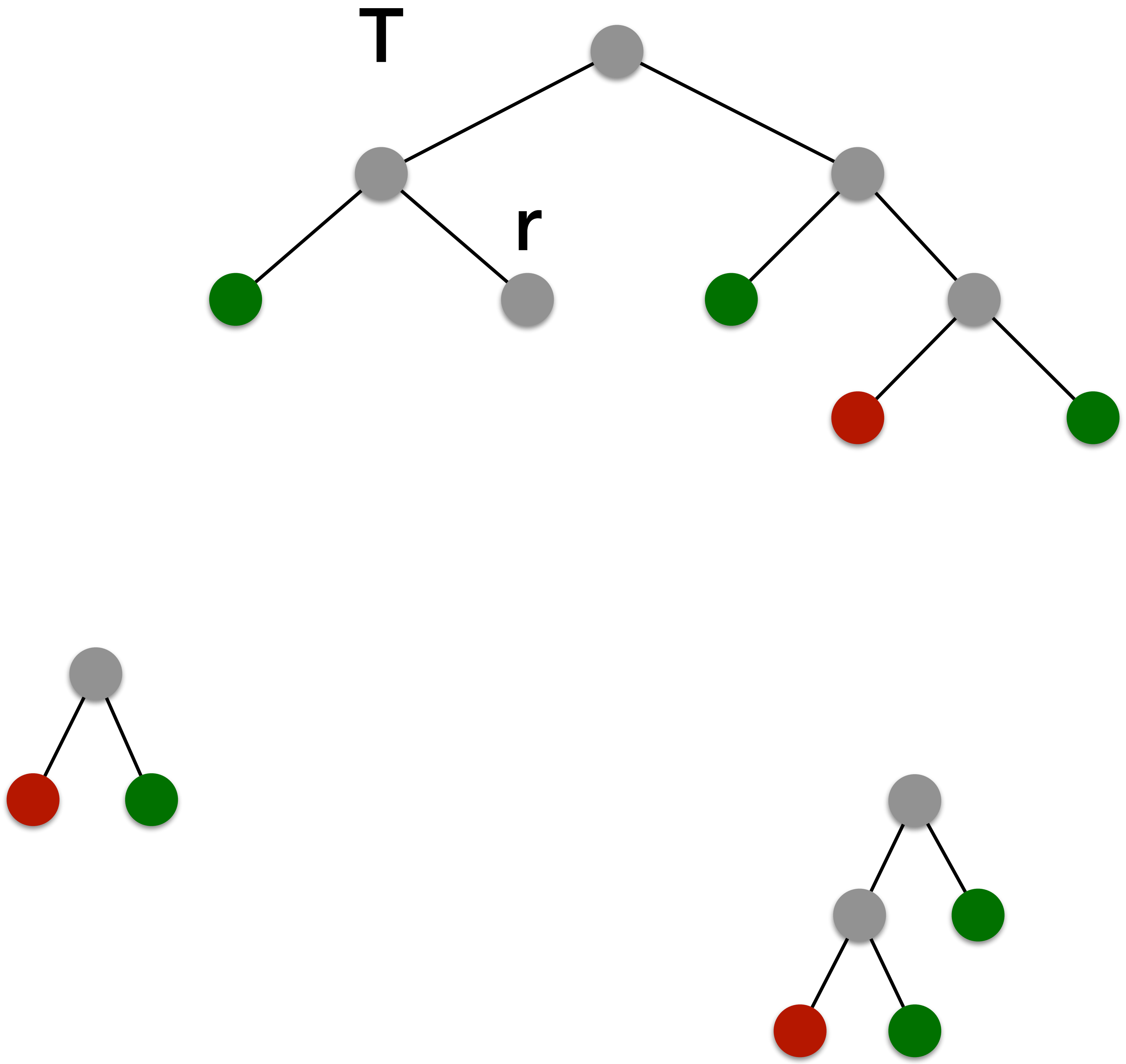
Improvement Step

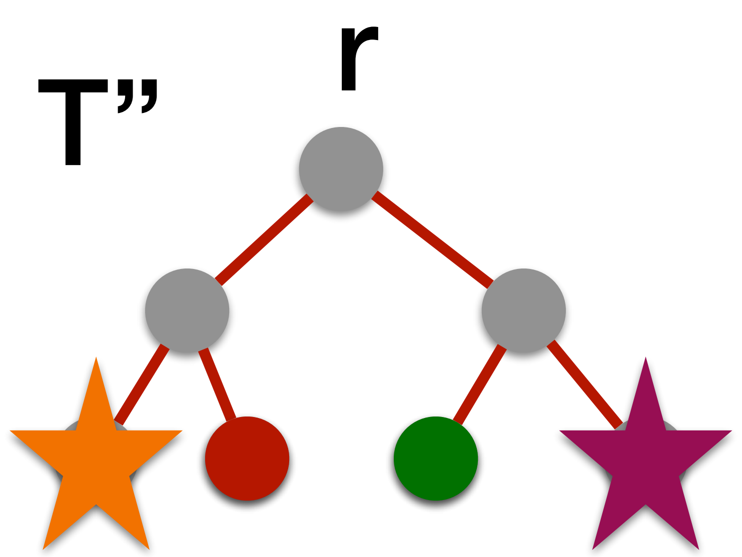




Find better tree T” for Local Instance using SAT

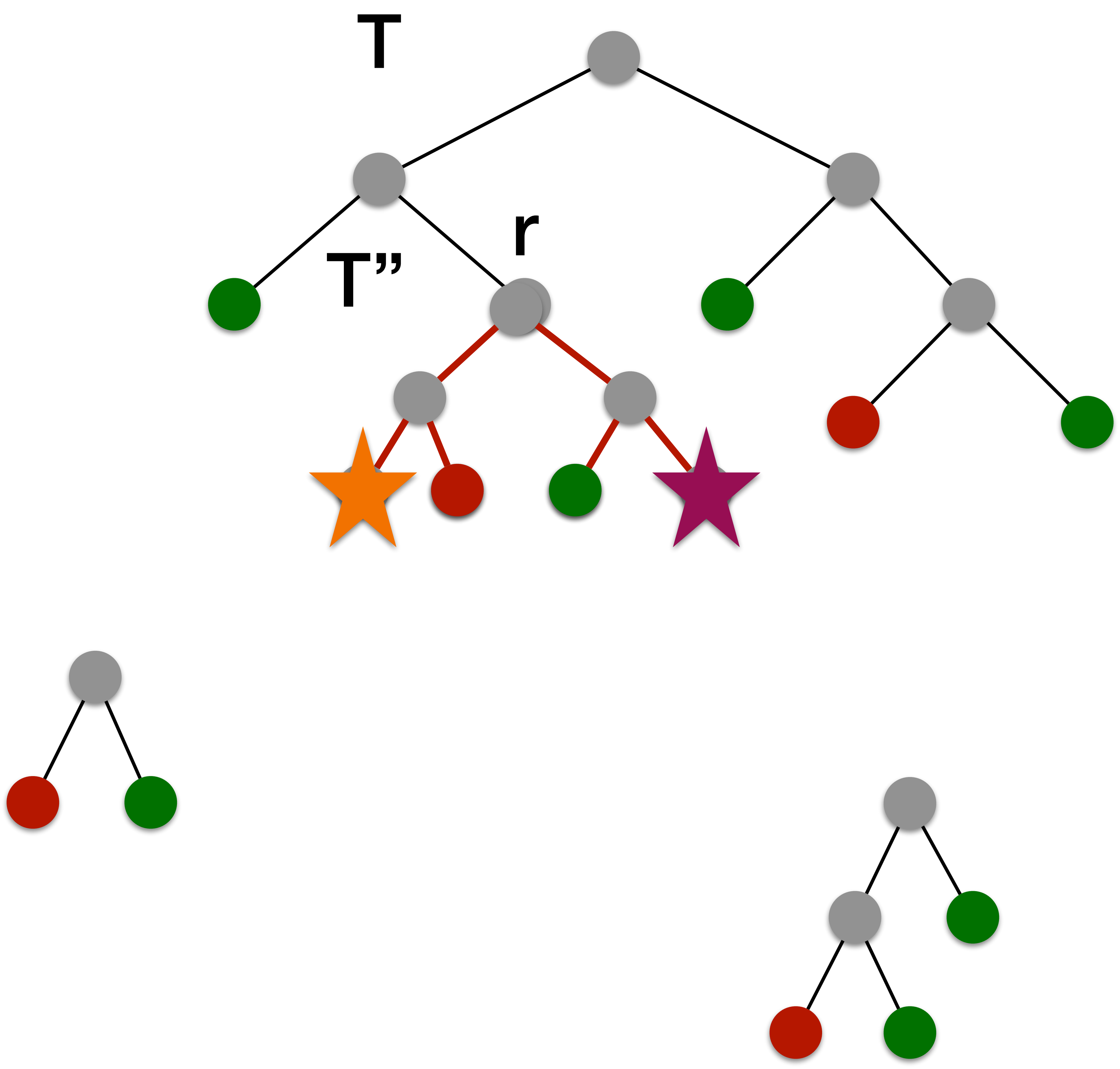
Improvement Step





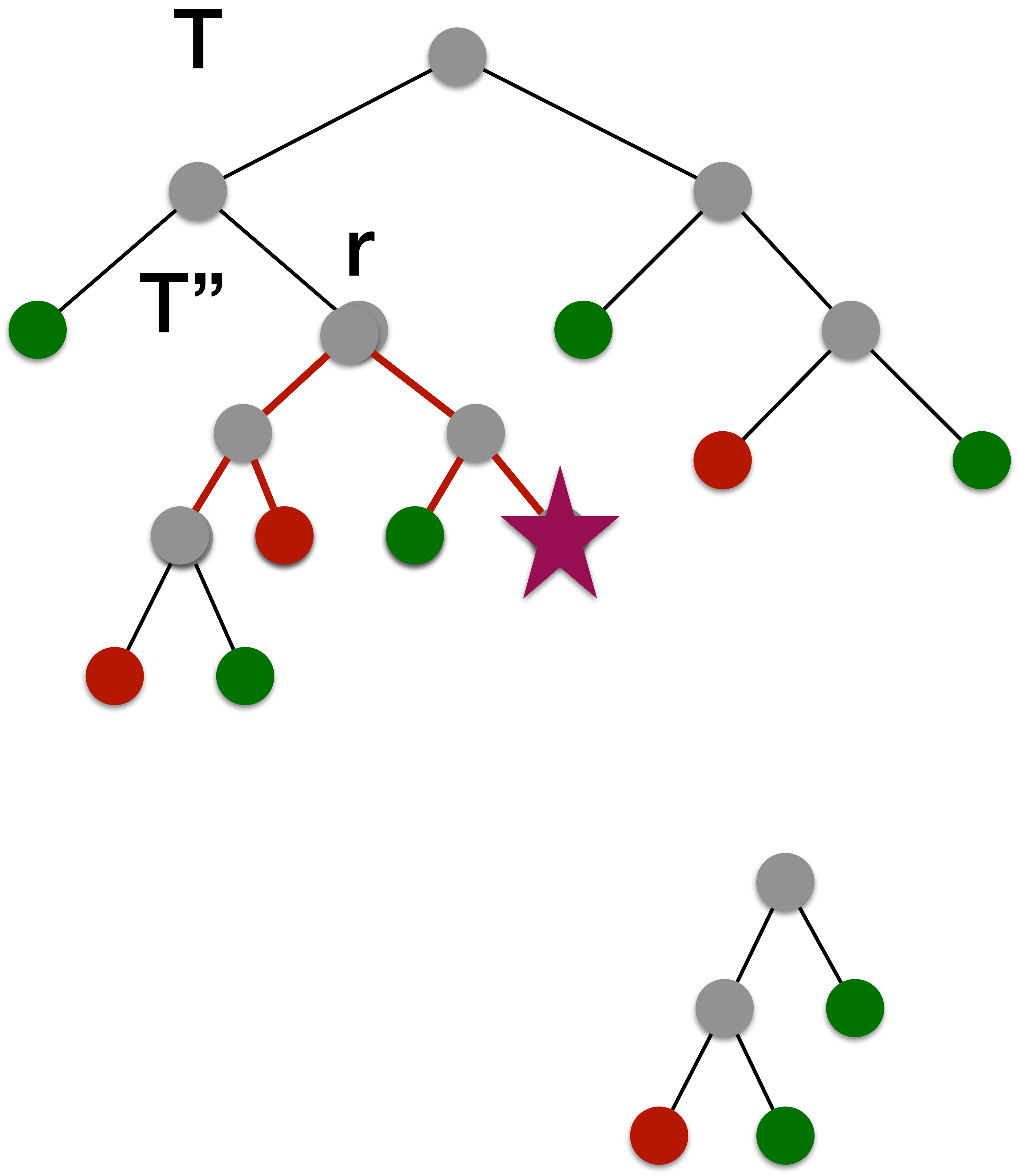
Find better tree T” for Local Instance using SAT

Improvement Step



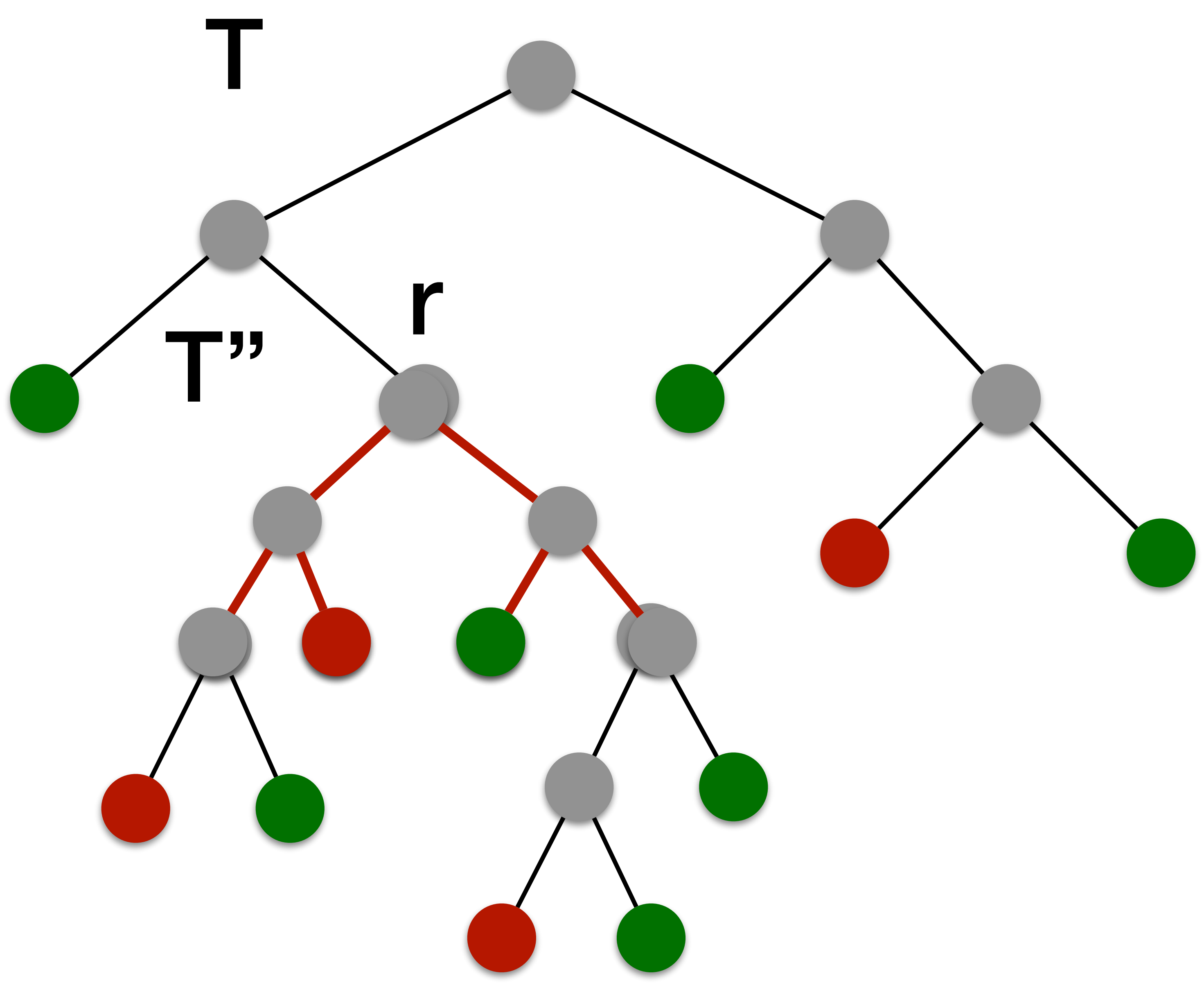
replace T’ with T”

Improvement Step



replace T’ with T”

Improvement Step



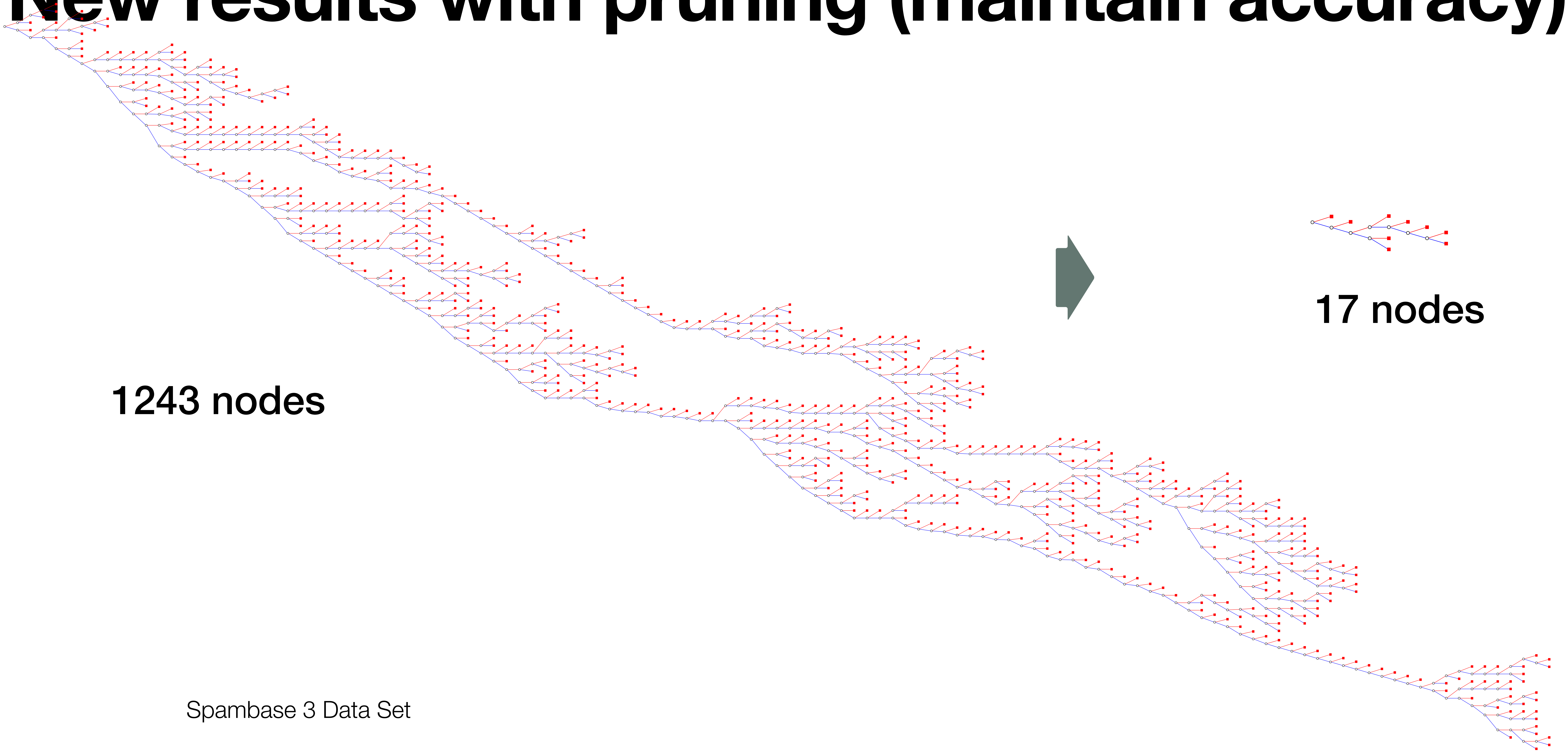
replace T’ with T”

Results

Instance			Weka		DT-SLIM	
Name	$\|F\|$	$\|E\|$	d	a	d	a
australian	1163	552	53.00	0.14	22.00	0.14
ccdefault	211	23955	96.60	0.71	80.00	0.71
haberman	92	240	71.20	0.66	63.80	0.62
hiv schilling	40	2617	18.80	0.81	11.80	0.79
hungarian	330	235	27.00	0.19	9.40	0.59
ida	2195	59998	61.00	1.00	51.00	1.00
objectivity	316	796	36.60	0.55	10.60	0.78

Comparison of depth and accuracy on selected instances before and after DT-SLIM

New results with pruning (maintain accuracy)



1243 nodes

17 nodes

Spambase 3 Data Set

Questions

- how to select the local instances?
- good tradeoff between budget and timeout dedicated to the local solver?
- Other applications?

Problem	**local solver**	**Paper**
Branchwidth	SAT	[Lodha, Ordyniak, Sz. (SAT'17, ToCL'19)]
Treewidth	SAT	[Fichte, Lodha, Sz. (SAT'17)]
Treedepth	MaxSAT	[Peruvemba Ramaswamy, Sz (CP'20)]
BN Structure Learning	MaxSAT	[Peruvemba Ramaswamy, Sz (AAAI'21)]
Decision Trees	SAT	[Schidler, Sz. (AAAI'21)]