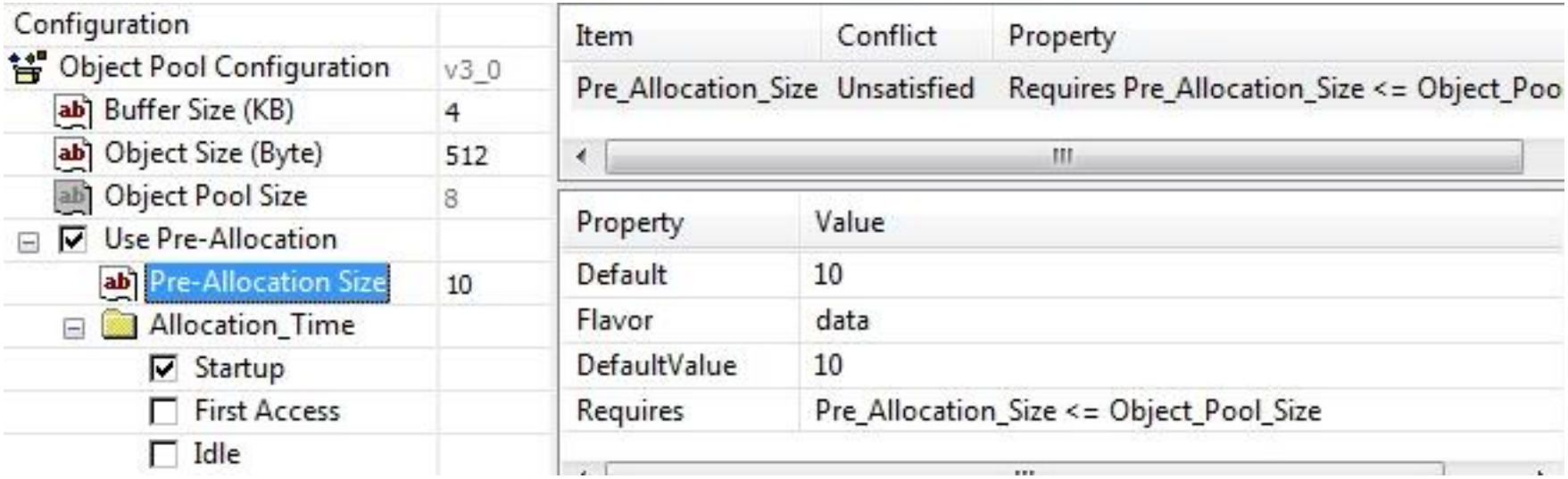
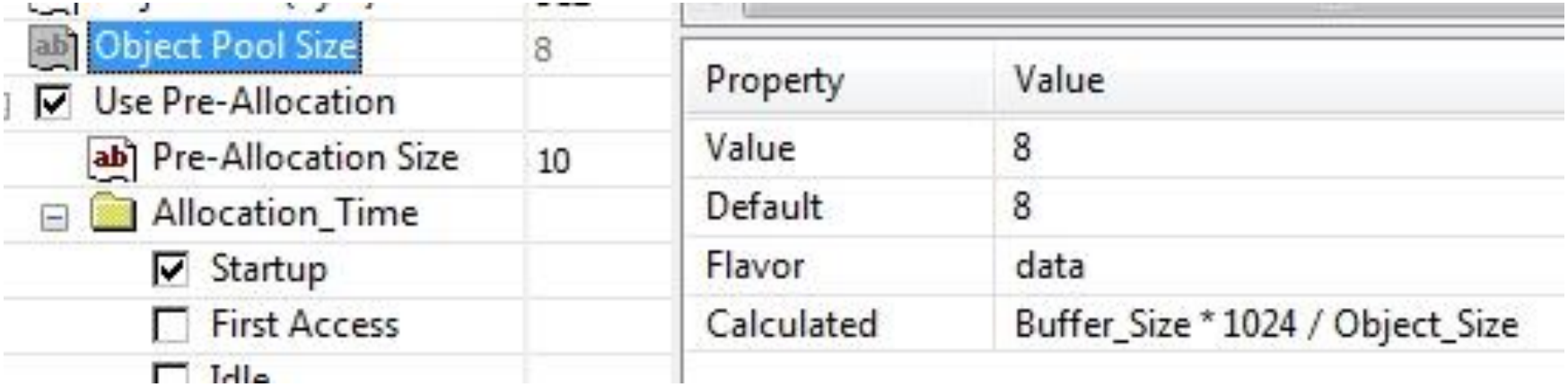
Efficient Range Fix Generation using HS-DAG

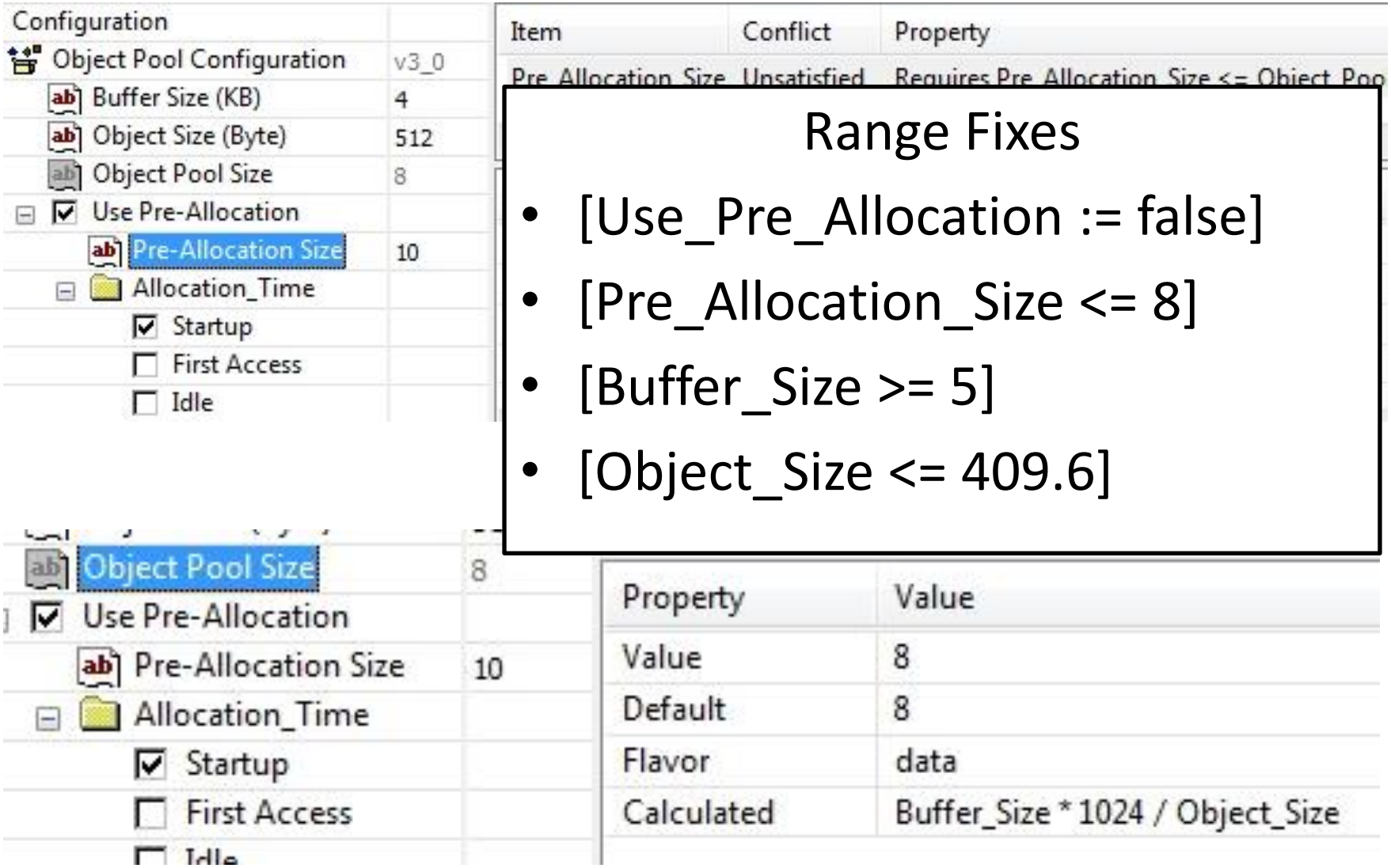
Yingfei Xiong, 2011

Content

- Review of Range Fixes
- Review of Reiter's theory of diagnosis and HS-DAG algorithm
- Combining the two and the experimental results


Configuration
Object Pool Configuration
v3_0
Buffer Size (KB)
4
Object Size (Byte)
512
Object Pool Size
8
Use Pre-Allocation
Pre-Allocation Size
10
Allocation_Time
Startup
First Access
Idle
Item
Conflict
Property
Pre_Allocation_Size
Unsatisfied
Requires Pre_Allocation_Size <= Object_Poo
Property
Value
Default
10
Flavor
data
DefaultValue
10
Requires
Pre_Allocation_Size <= Object_Pool_Size


Object Pool Size
8
Use Pre-Allocation
Pre-Allocation Size
10
Allocation_Time
Startup
First Access
Idle
Property
Value
Value
8
Default
8
Flavor
data
Calculated
Buffer_Size * 1024 / Object_Size


Configuration
Object Pool Configuration
v3_0
Buffer Size (KB)
4
Object Size (Byte)
512
Object Pool Size
8
Use Pre-Allocation
Pre-Allocation Size
10
Allocation_Time
Startup
First Access
Idle
Item
Conflict
Property
Pre_Allocation_Size
Unsatisfied
Requires Pre_Allocation_Size <= Object_Poo
Range Fixes
• [Use_Pre_Allocation := false]
• [Pre_Allocation_Size <= 8]
• [Buffer_Size >= 5]
• [Object_Size <= 409.6]
Object Pool Size
8
Use Pre-Allocation
Pre-Allocation Size
10
Allocation_Time
Startup
First Access
Idle
Property
Value
Value
8
Default
8
Flavor
data
Calculated
Buffer_Size * 1024 / Object_Size

Correctness

- Any value represented by a range fix will satisfy the constraint
- Example
  - Constraint: $a \geq b$
  - Configuration: $\{a \mapsto 1, b \mapsto 10\}$
- Range Fixes
  - $[a\colon\ a\ >\ 8]$ 
  - $[a\colon\ a \geq 10]$ 

Strict Minimality

- There is no way to change a subset of variables to fix the inconsistency
- Example
  - Constraint: $a \geq b$
  - Configuration: $\{a \mapsto 1, b \mapsto 10\}$
- Range Fixes
  - $[(a, b): a \geq b]$ 
  - $[a: a \geq 10]$ 

Completeness

- A range fix represents the maximal ranges over the variables
- Example
  - Constraint: $a \geq b$
  - Configuration: $\{a \mapsto 1, b \mapsto 10\}$
- Range Fixes
  - $[a: a > 11]$ 
  - $[a: a \geq 10]$ 

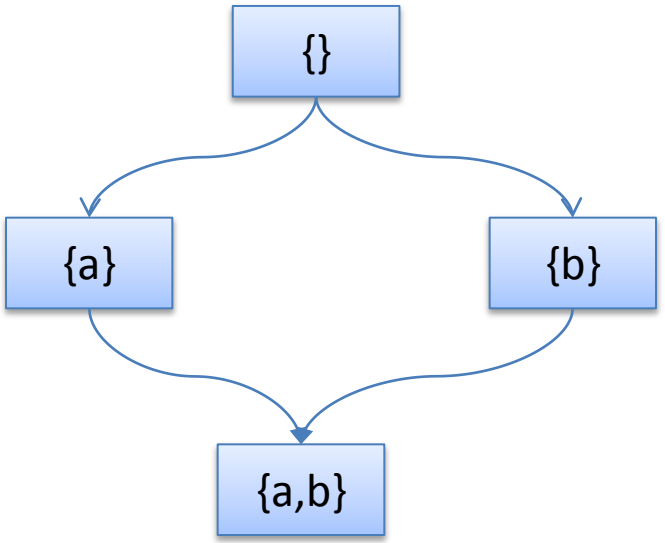
Range Fix Generator

- Input:
  - A constraint
  - A configuration violating the constraints
- Output:
  - all correct, minimal, and complete range fixes

Basic Algorithm: Exhaustive Search

- Iterate all possible variable combinations

  - {}: a >=b /\ a = 1 /\ b = 10
    - unsat
  - {a}: a >= b /\ b=10
    - sat
    - replacing b by 10: a >= 10
  - {b}: a >= b /\ a = 1
    - sat
    - replacing a by 1: b <= 1
  - {a,b}: not tried



$a \geq b$

$\{a \mapsto 1, b \mapsto 10\}$

Improvements of Exhaustive Search

- Apply binary search
  - unsat: remove all ancestors
  - sat: remove all descendants
- Separate Boolean and non-Boolean variables
  - Boolean fixes can be directly generated by converting to DNF

Core problem of generating range fixes

- How to get the minimal sets of variables that must be changed?

(Simplified) Reiter's Theory of Diagnosis

- An abnormal system $(COMPONENTS, OBS)$ is two sets of constraints that are inconsistent (their conjunction is unsatisfiable).
- A diagnosis is a minimal set $\Delta \in COMPONENTS$ such that $(COMPONENTS - \Delta) \cup OBS$ is consistent

Relation to Range Fix Generation

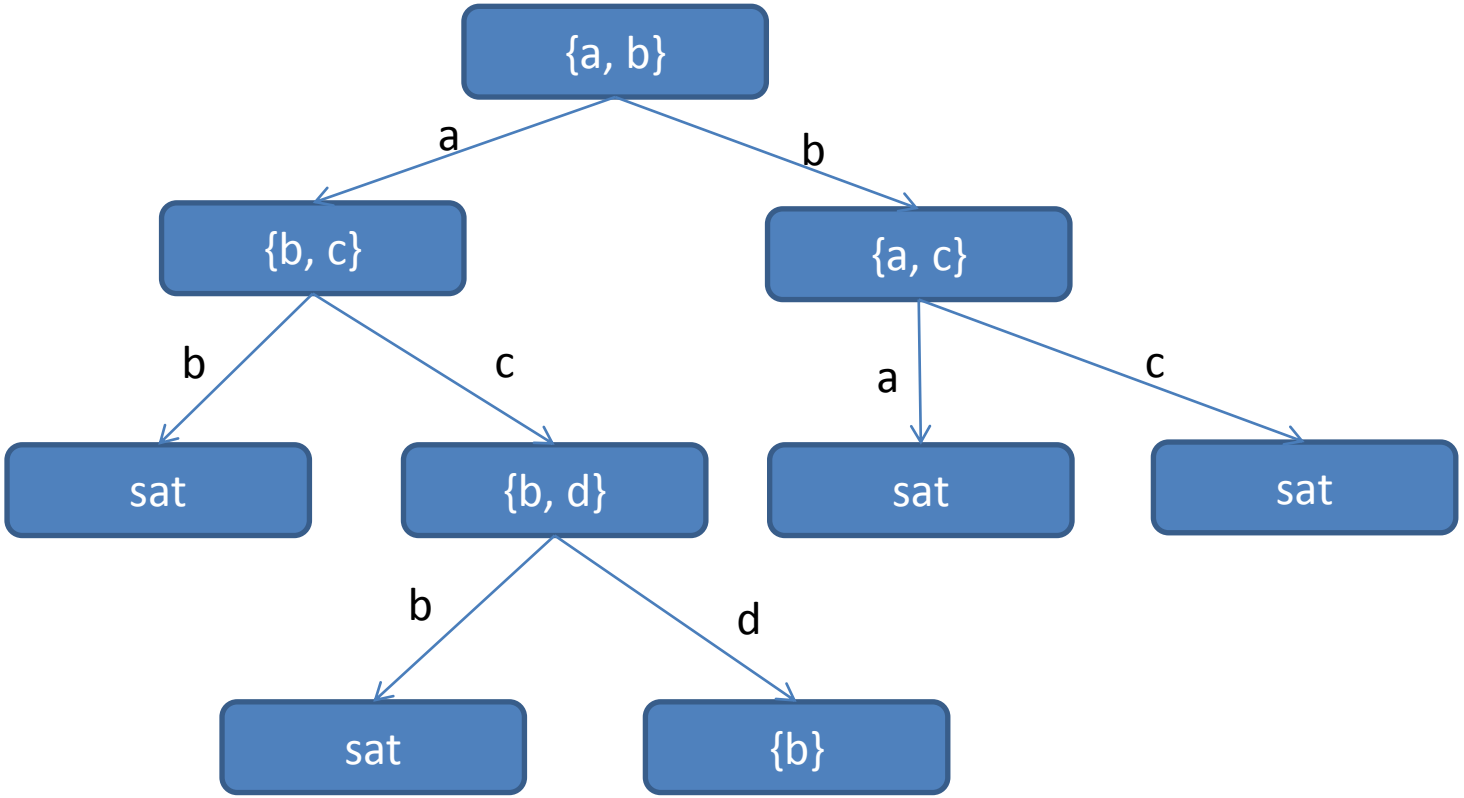
- Example
  - Constraint: $a \geq b$
  - Configuration: $\{a \mapsto 1, b \mapsto 10\}$
- Convert to Reiter's system
  - OBS: $\{a \geq b,\ p_a \rightarrow a = 1,\ p_b \rightarrow b = 1\}$
  - COMPONENT: $\{p_a, p_b\}$
- A diagnosis is a set of variables that must be changed

Core-Guided Diagnosis Generation (HS-DAG)

- Let C be a set of sets. A hitting set for C is a set $H \subseteq \bigcup_{S \in C} S$ such that $H \cap S \neq \emptyset$ for any $S \in C$. A hitting set is minimal iff no proper subset of it is a hitting set.
- Theorem:
  $\Delta \subseteq COMPONENTS$ is a diagnosis for $(COMPONENTS, OBS)$ such that Δ is a minimal hitting set for the collection of unsatisfiable cores of $COMPONENTS \cup OBS$

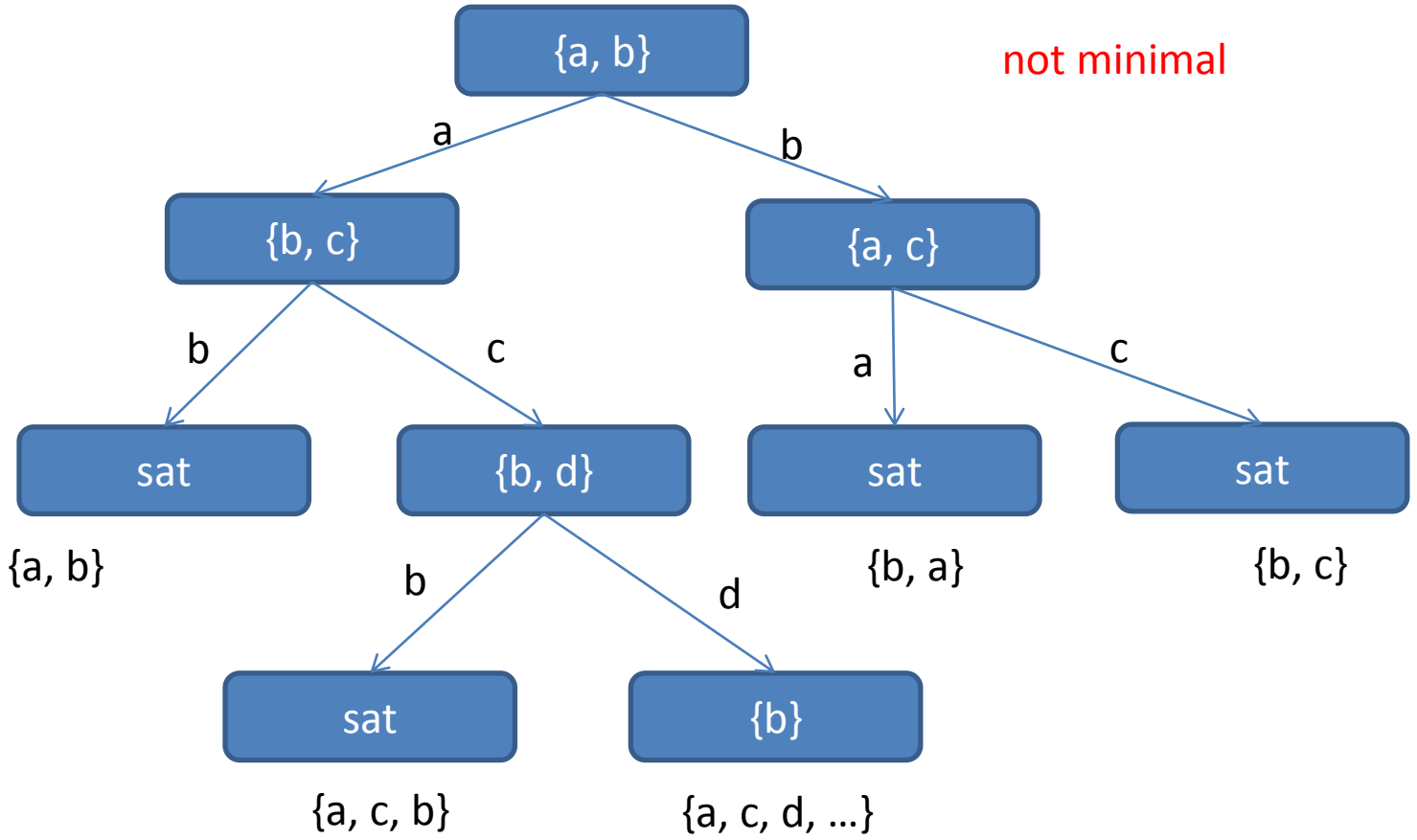
Core-Guided Diagnosis Generation (HS-DAG)

- cores: {{a, b}, {b, c}, {a, c}, {a, b}, {b, d}, {b}}

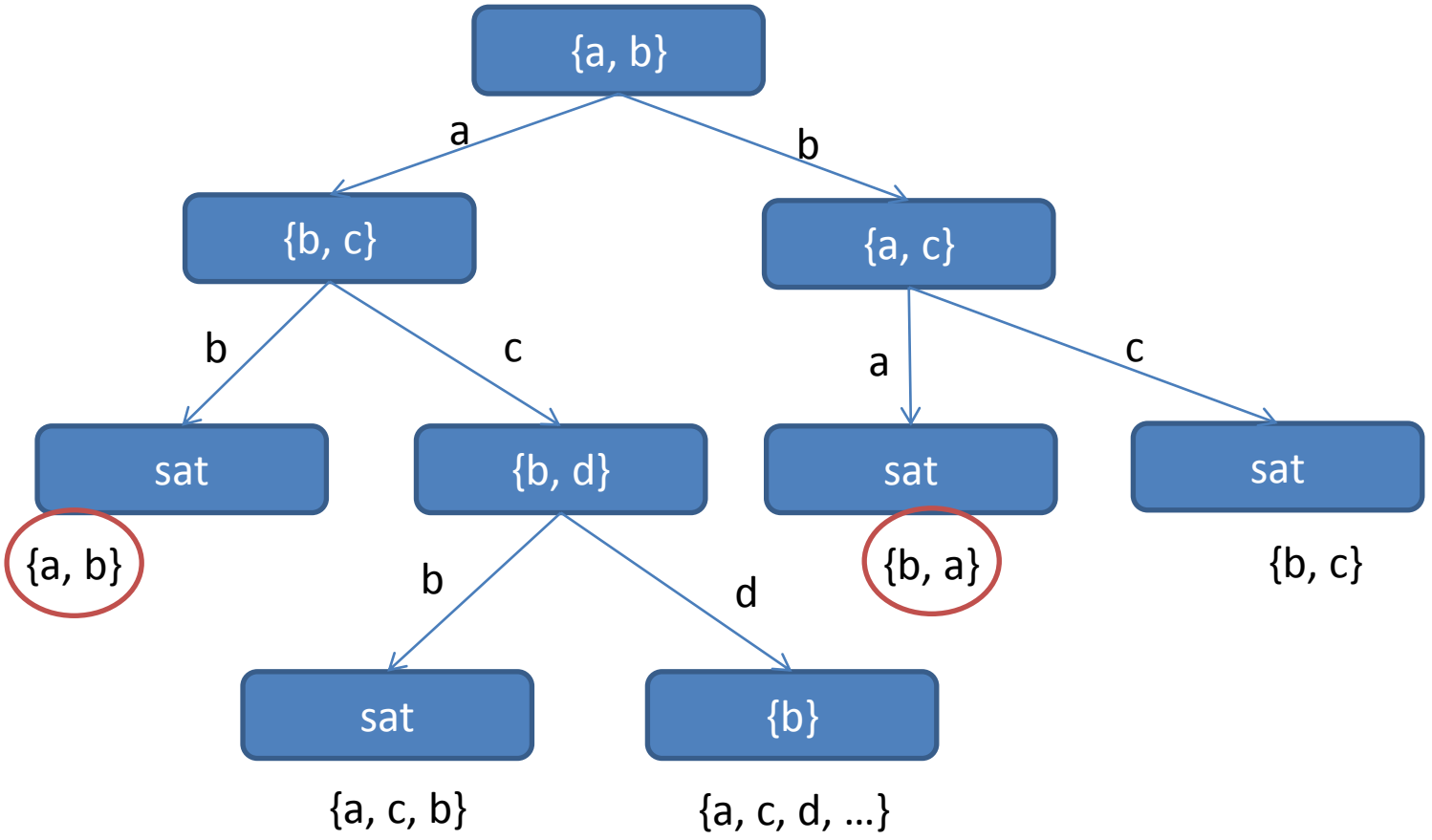


Core-Guided Diagnosis Generation

- cores: {{a, b}, {b, c}, {a, c}, {a, b}, {b, d}, {b}}

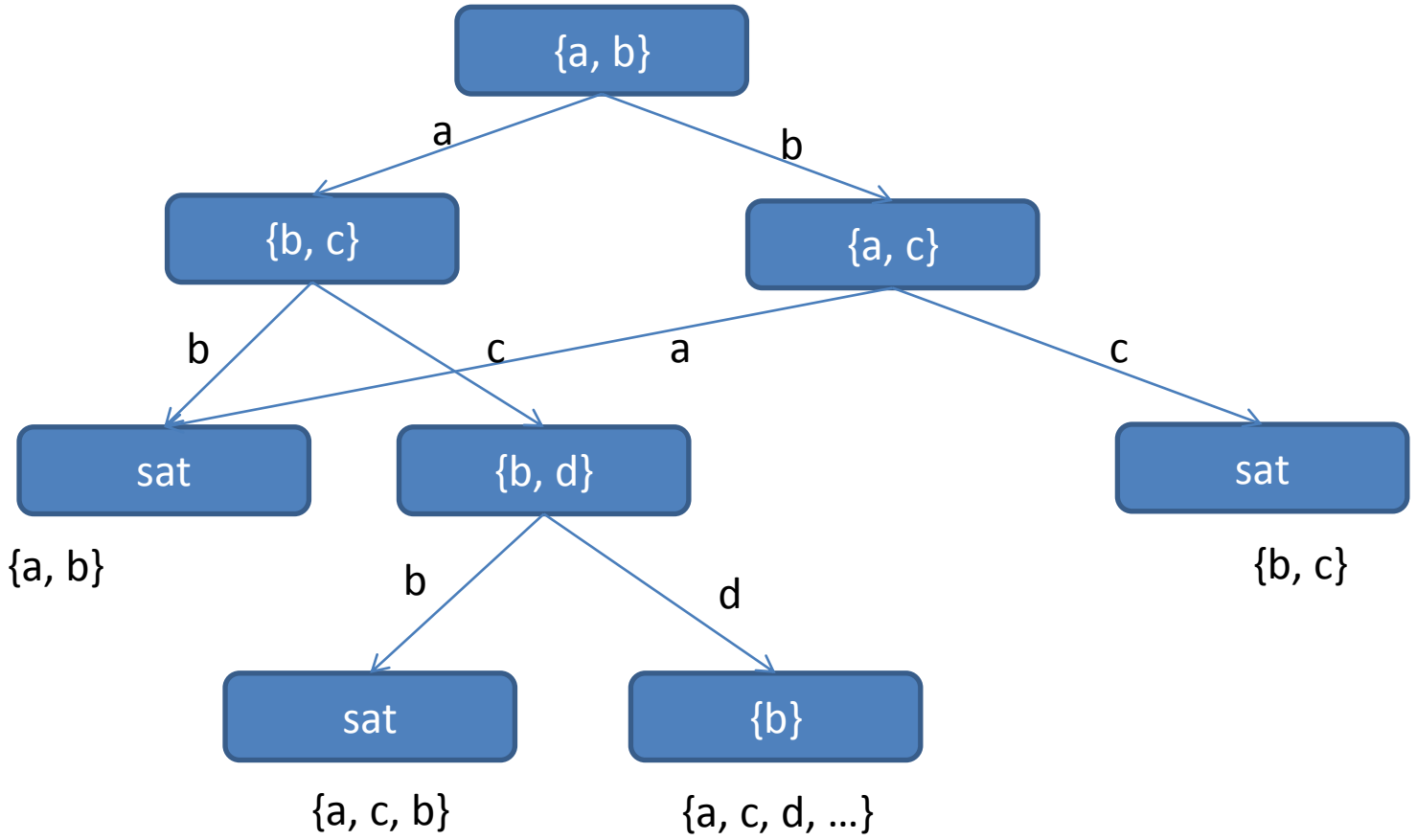


Problem 1: Duplicates

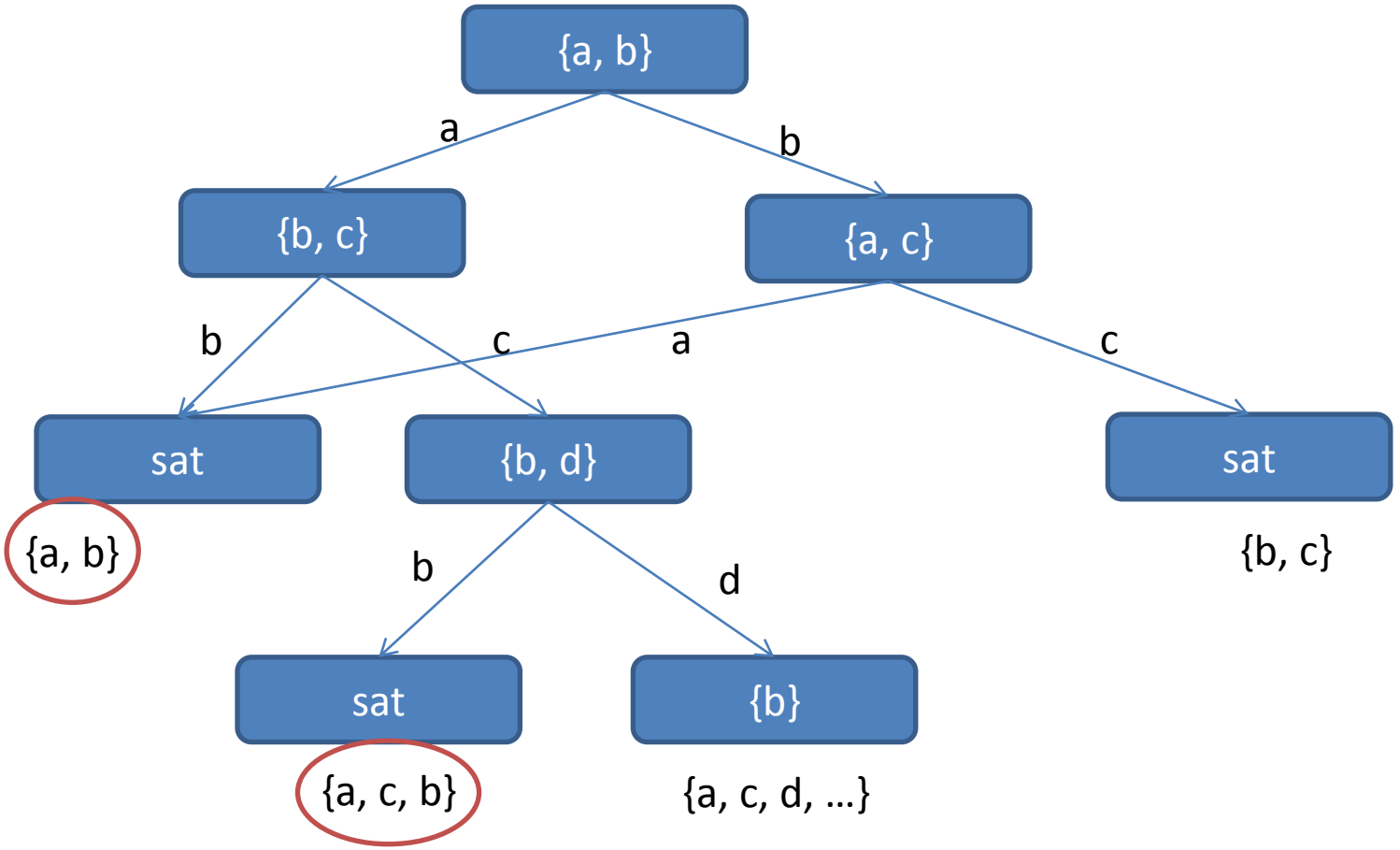


Solution: Reusing nodes

- Reuse instead of creating a new node if there exists a node such that the paths are labeled the same

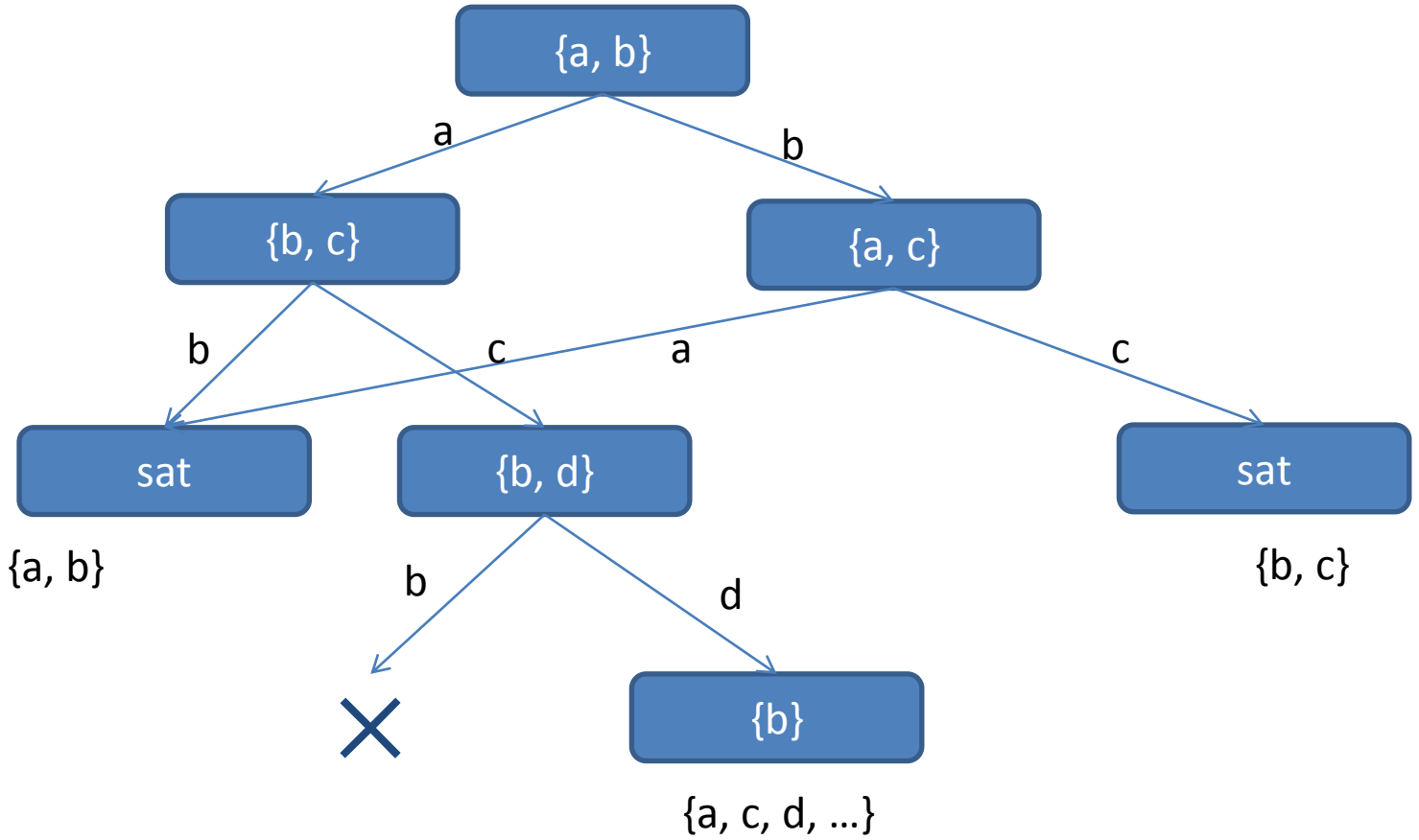


Problem 2: Non-minimal path

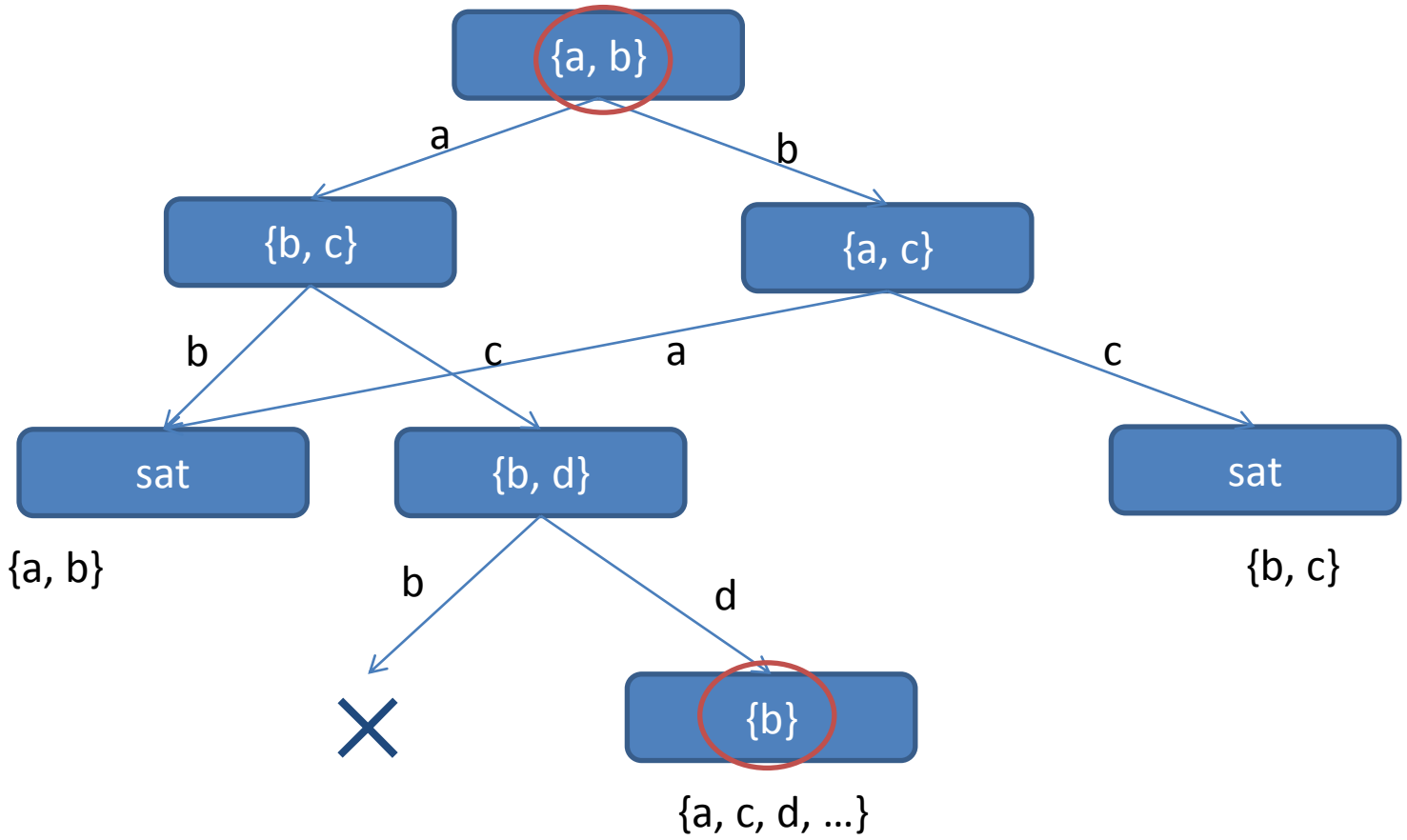


Solution: Close Node

- Close a node (not exploring futher) when its path is not minimal

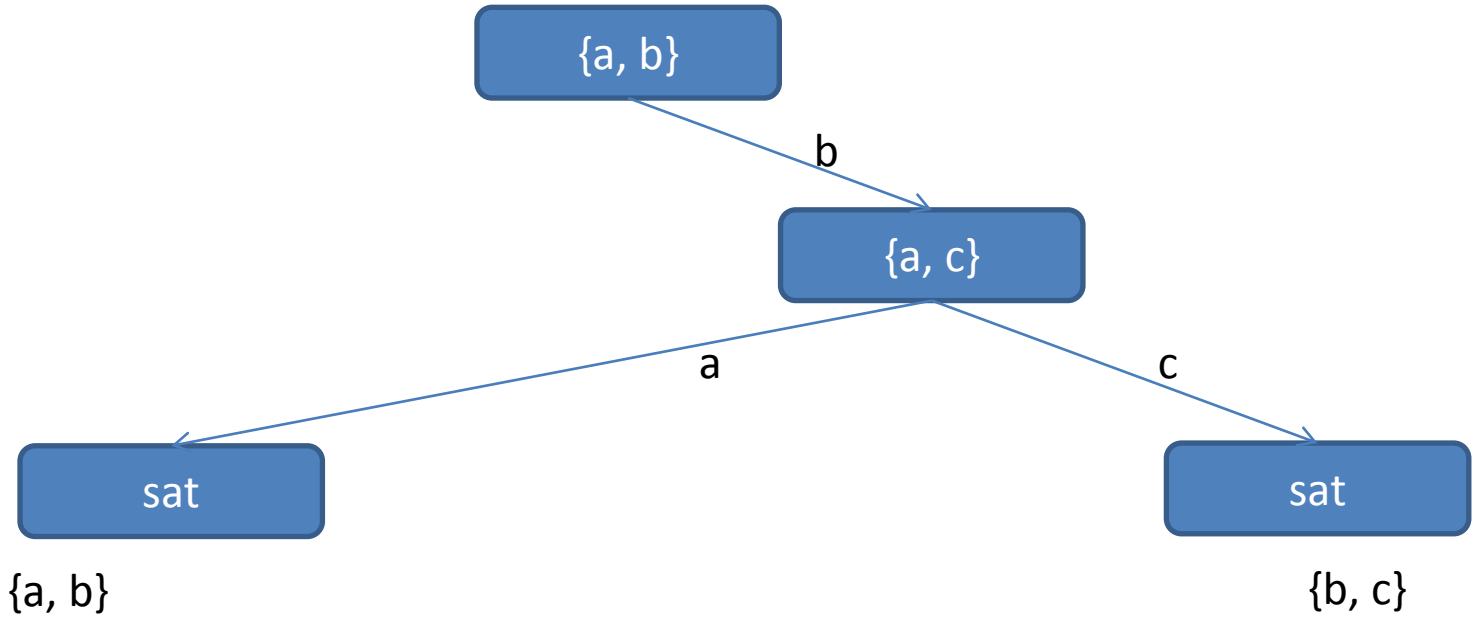


Problem 3: Cores are not minimal



Solution: Pruning

- Prune the branch caused by non-minimal cores



Comparison of Algorithms

- Improved Exhaustive Search:
  - number of non-Boolean variables ↑ implies
  - execution time ↑
- HS-DAG:
  - number of cores ↑ implies
  - execution time ↑

Experimental Result (for one constraint)

	Min	Max	Avg	Med
Execution Time(ms)	15	33	17.98	17
Number of Fixes	1	4	1.95	2
Max Size of Fixes	1	3	1.32	1
Σ Size of Fixes	2	12	2.34	2

Interaction of Constraints

- Fixes for one constraint should not violate other satisfied constraints
- Let c be the violated constraint, c1, c2,…, cn be the satisfied constraints
- generate fixes for
  - c /\ c1 /\ c2 /\ … /\ cn

Experimental Result (Interaction Considered)

	Min	Max	Avg	Med
Execution Time	289	2201	444.10	373
Number of Fixes	1	13	2.0	2
Max Size of Fixes	1	12	2.9	2
Σ Size of Fixes	2	74	5.78	3