SAT Solvers and Search-Based Client Applications

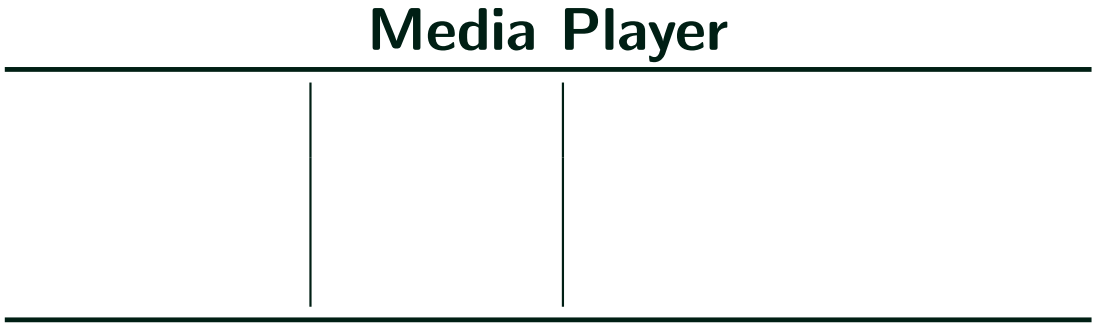
Brady J. Garvin

University of Nebraska–Lincoln

September 11, 2009

Outline

Highly-Configurable Systems



Highly-Configurable Systems

Media Player

Encoding		
MPEG		
RAW		

Highly-Configurable Systems

Media Player

Encoding	*Format*
MPEG	AUDIO
RAW	VIDEO

Highly-Configurable Systems

Media Player

Encoding	*Format*	*Closed-Captioning*
MPEG	AUDIO	YES
RAW	VIDEO	NO

Highly-Configurable Systems

Media Player

Encoding	*Format*	*Closed-Captioning*
MPEG	AUDIO	YES
RAW	VIDEO	NO

Closed-Captioning=YES requires *Format*=VIDEO

Highly-Configurable Systems

Media Player

Encoding	*Format*	*Closed-Captioning*
MPEG	AUDIO	YES
RAW	VIDEO	NO

Closed-Captioning=YES requires *Format*=VIDEO

(No fault)

Highly-Configurable Systems

Media Player

Encoding	*Format*	*Closed-Captioning*
MPEG	AUDIO	YES
RAW	VIDEO	NO

Closed-Captioning=YES requires *Format*=VIDEO

(Interaction fault)

Highly-Configurable Systems

Media Player

Encoding	*Format*	*Closed-Captioning*
MPEG	AUDIO	YES
RAW	VIDEO	NO

Closed-Captioning=YES requires *Format*=VIDEO

(Constraint violation)

Covering Arrays

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	YES
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	YES
4.	RAW	VIDEO	NO

Covering Arrays

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	YES
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	YES
4.	RAW	VIDEO	NO

Covering Arrays

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	YES
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	YES
4.	RAW	VIDEO	NO

Covering Arrays

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	YES
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	YES
4.	RAW	VIDEO	NO

Constrained Covering Arrays

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	YES
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

Outline

Element Replacement

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

(Missing 2 pairs: [MPEG,VIDEO] and [MPEG,YES])

Element Replacement

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

(Missing 1 pair: [MPEG,YES])

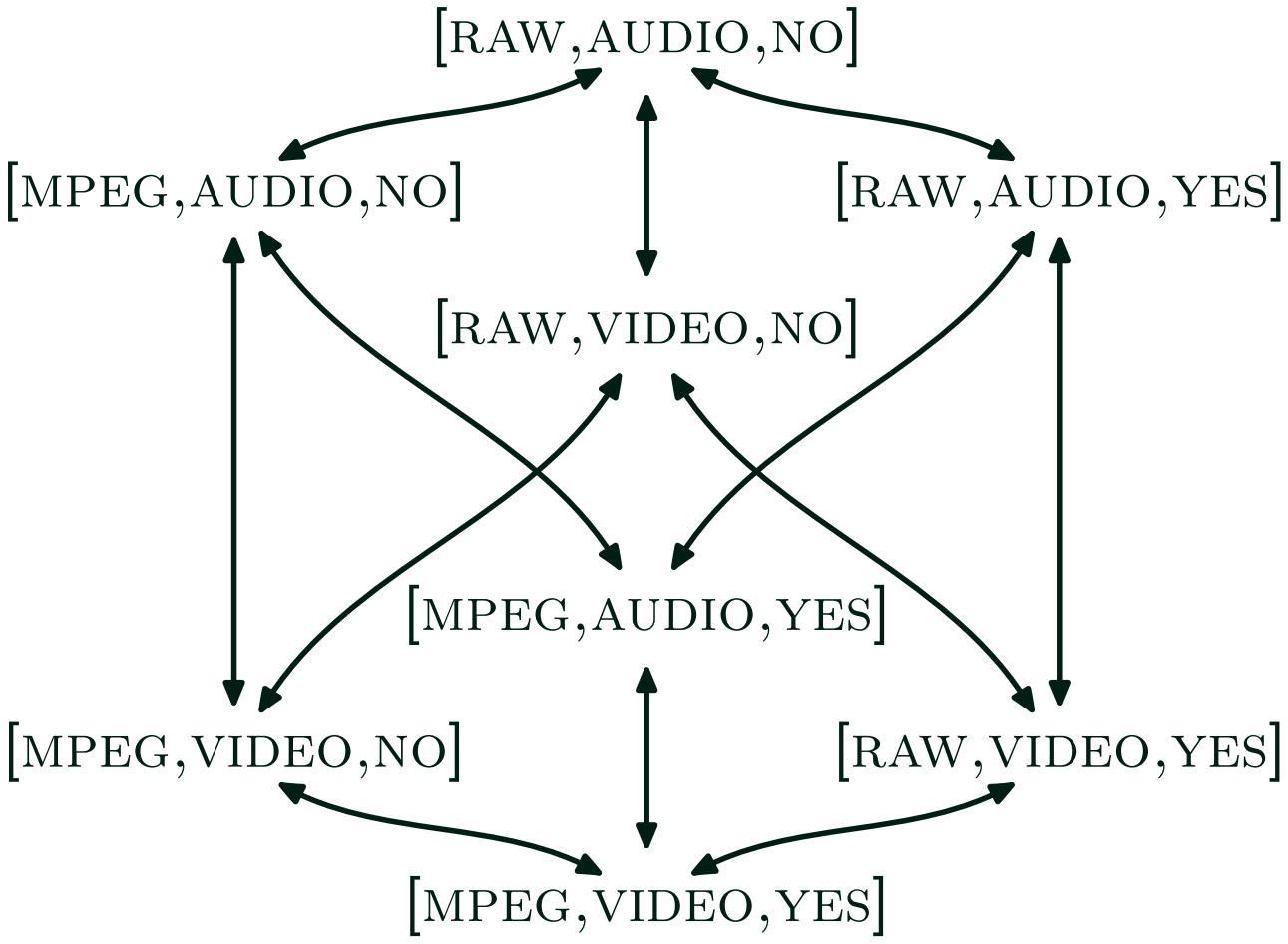
Element Replacement

	Encoding	*Format*	*Closed-Captioning*
1.	MPEG	VIDEO	NO
2.	MPEG	AUDIO	YES
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

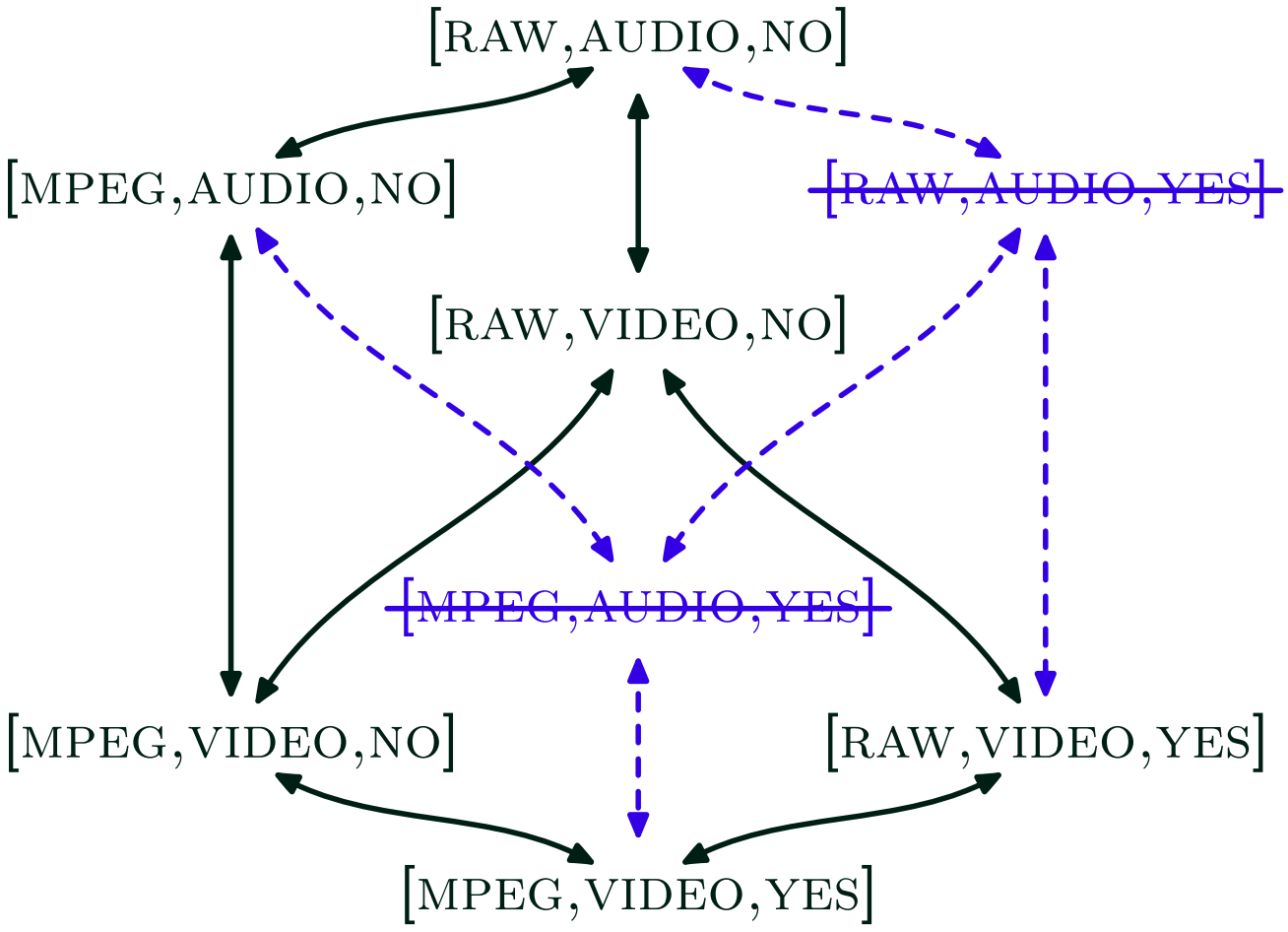
$$\text{VIDEO} \vee \neg\text{YES}$$

(Rejected by death penalty)

Element Replacement



Element Replacement



t-set Replacement

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

(Missing 2 pairs: [MPEG,VIDEO] and [MPEG,YES])

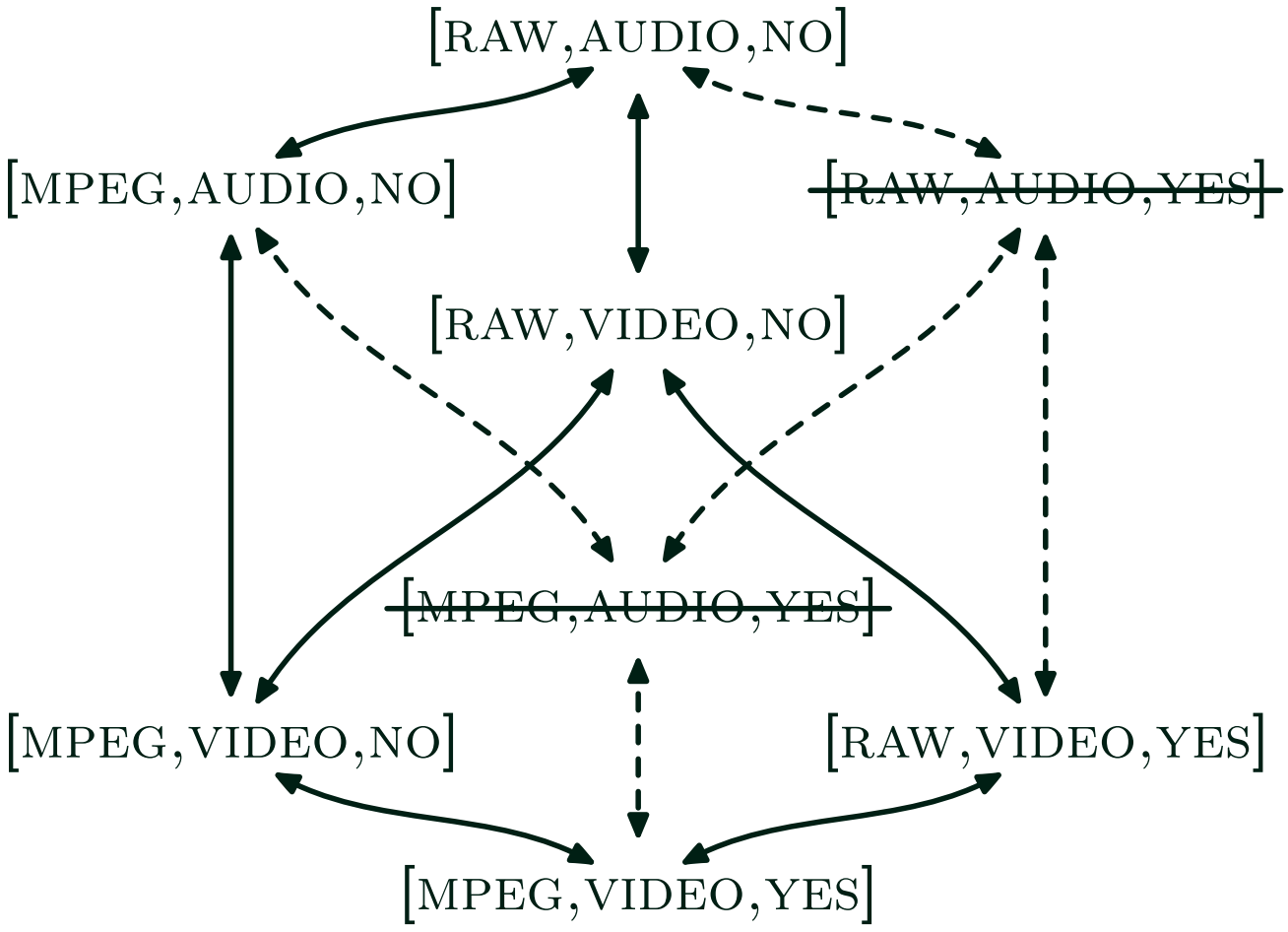
t-set Replacement

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	MPEG	VIDEO	YES
5.	RAW	VIDEO	YES

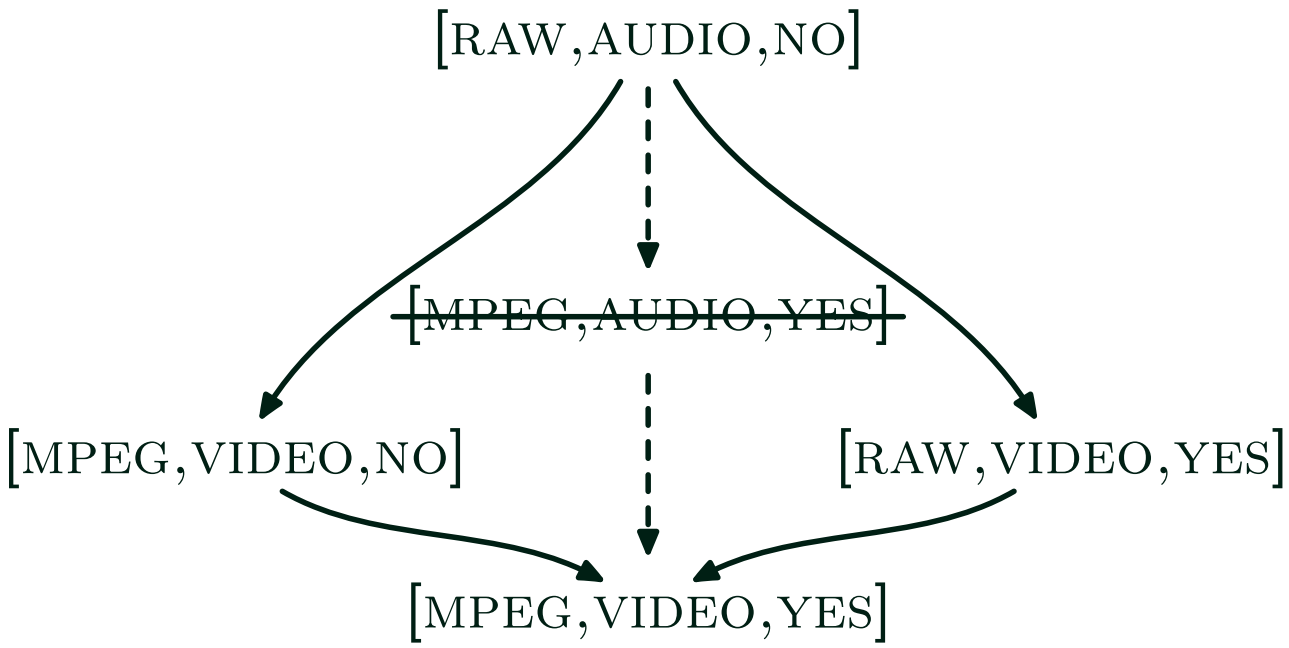
$$\text{VIDEO} \vee \neg\text{YES}$$

(Missing 0 pairs)

t-set Replacement



t-set Replacement



Outline

t-set Tracking

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

(1 unique pair: [RAW,AUDIO])
(Missing 2 pairs: [MPEG,VIDEO] and [MPEG,YES])

Using t-set Tracking

- If we restrict our search to one row, we want to maximize the coverage of unique and missing t-sets.
- In this case we should find a row to satisfy $\textsc{video} \vee \neg\textsc{yes}$ while maximizing the number of true terms in $\{(\textsc{raw} \wedge \textsc{audio}), (\textsc{mpeg} \wedge \textsc{video}), (\textsc{mpeg} \wedge \textsc{yes})\}$

Using t-set Tracking

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	RAW	AUDIO	NO
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

$$\text{VIDEO} \vee \neg\text{YES}$$

(1 unique pair: [RAW,AUDIO])
(Missing 2 pairs: [MPEG,VIDEO] and [MPEG,YES])

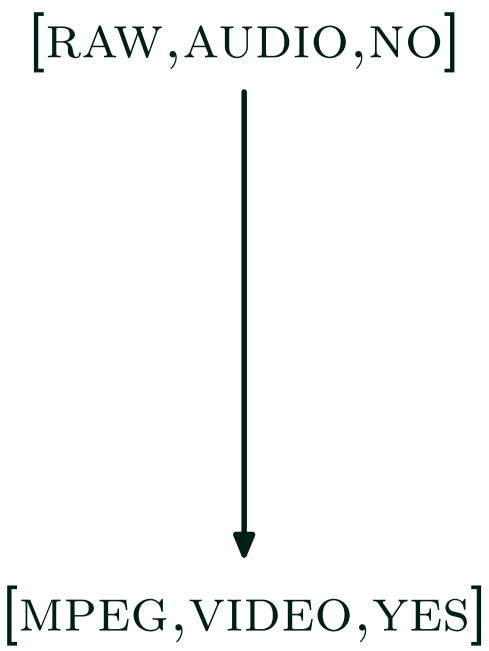
Using t-set Tracking

	Encoding	*Format*	*Closed-Captioning*
1.	RAW	VIDEO	NO
2.	MPEG	AUDIO	NO
3.	MPEG	VIDEO	YES
4.	RAW	VIDEO	NO
5.	RAW	VIDEO	YES

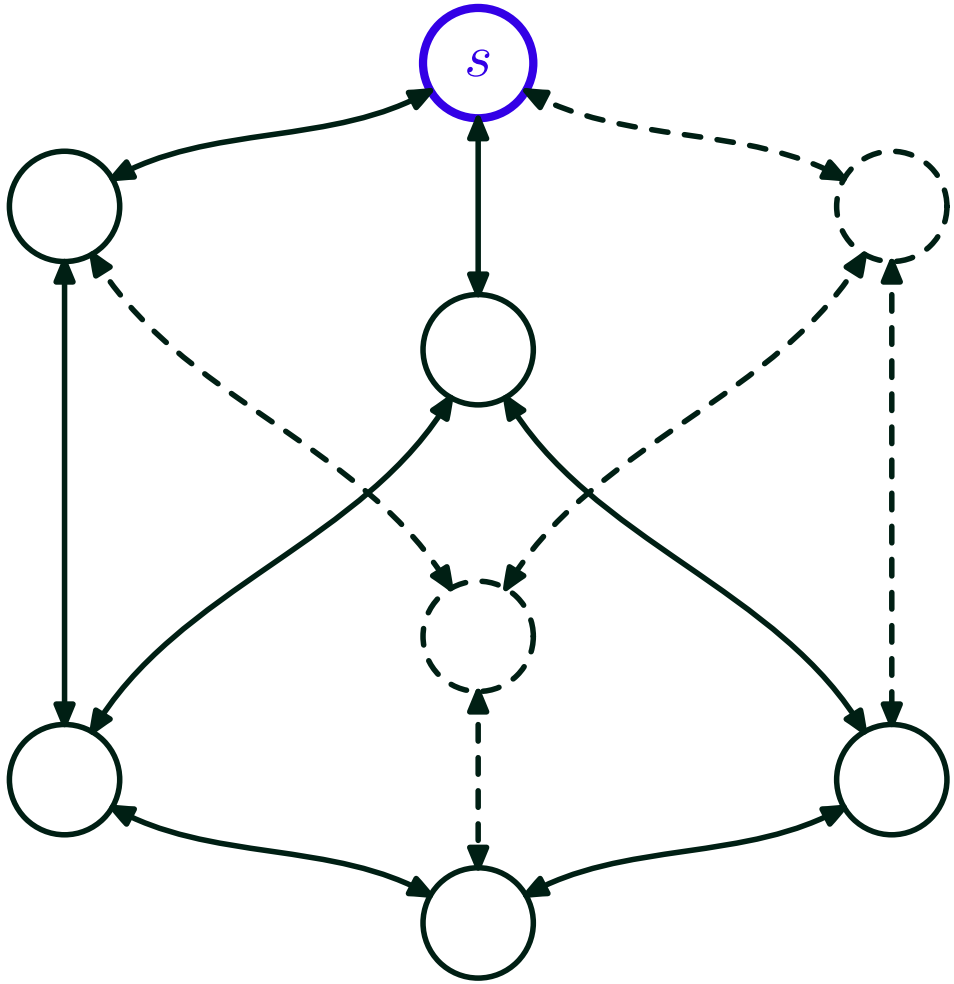
$$\text{VIDEO} \vee \neg\text{YES}$$

(2 unique pairs: [MPEG,VIDEO] and [MPEG,YES])
(Missing 1 pair: [RAW,AUDIO])

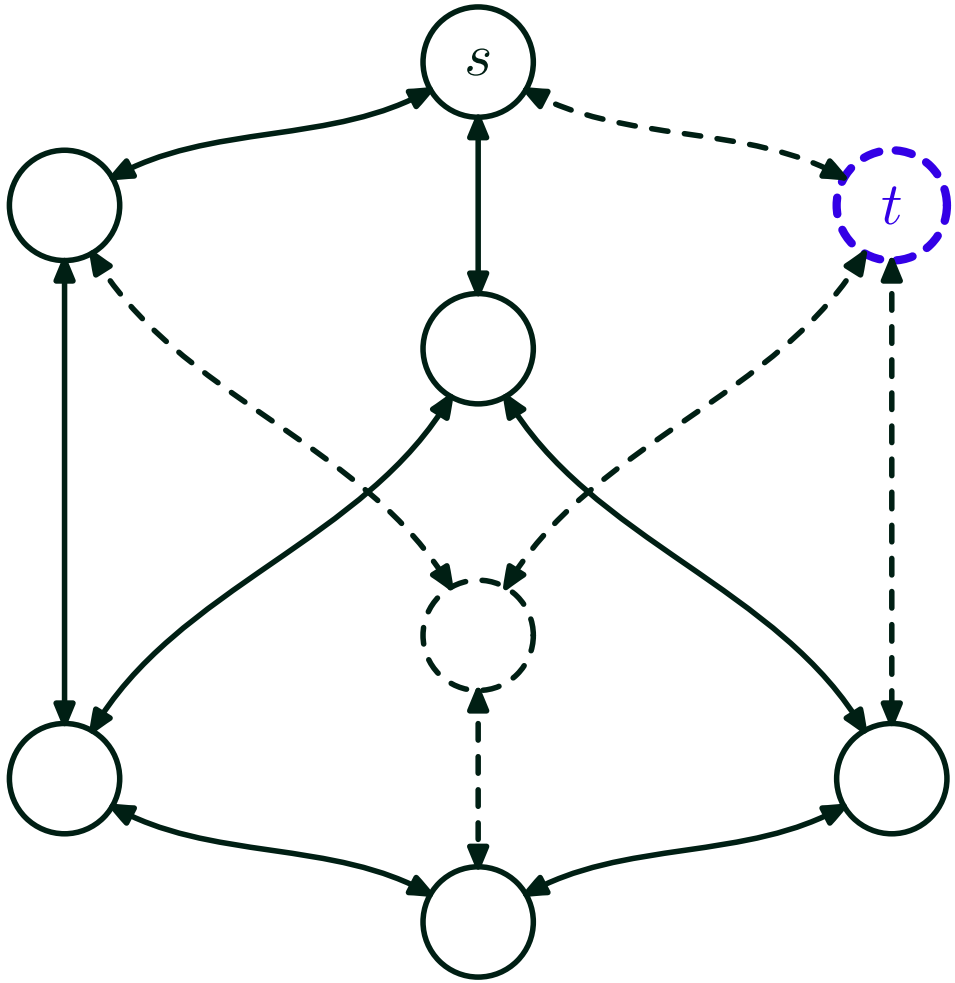
Using t-set Tracking



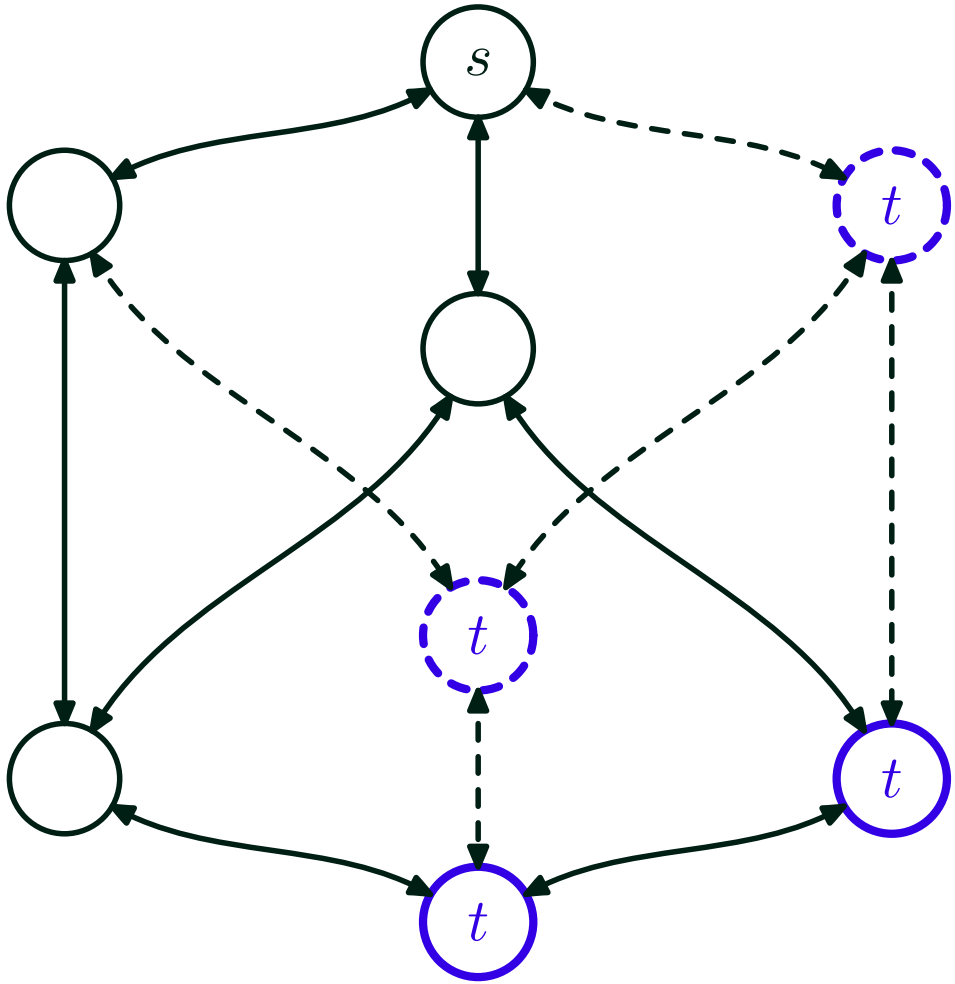
Other Constrained Searches



Other Constrained Searches



Other Constrained Searches



Possible Solutions

- Pseudo-Boolean Encoding
- Pure SAT Encoding
- SAT Encoding with an Objective Function

Possible Solutions

- In DPLL we can maximize our objective function with very little change to the code:
  1. Define a variable τ_i for each unique or missing t-set.
  2. Backtrack whenever the number of τ_is assigned true cannot exceed a given lower bound.
  3. Repeatedly search for a solution, increasing the lower bound until the problem becomes UNSAT.

Possible Solutions

- There are also performance reasons to change the row as little as possible.
- That isn't accounted for in the candidate solutions.

Outline

Conclusions

- While improving a meta-heuristic constrained CIT sample generator we discovered that almost all of the challenges we faced stemmed from constraints.
- We improved performance using a SAT solver normally.
- But with a simple objective function we could do better.
- The same applies to other constrained searches.

Jump to Slide (1–40)