Decision Procedures in First Order Logic

Decision Procedures for Equality Logic

Part III – for Equality Logic and Uninterpreted Functions

- Algorithm I – From Equality to Propositional Logic
  - Adding transitivity constraints
  - Making the graph chordal
  - An improved procedure: consider polarity
- Algorithm II – Range-Allocation
  - What is the small-model property?
  - Finding a small adequate range (domain) to each variable
  - Reducing to Propositional Logic

Summary

1. Let E denote the set of equality predicates appearing in ϕ^E . Derive a Boolean formula ϕ_{enc} by replacing each equality predicate $(x_i = x_j)$ in E with a new Boolean variable $e_{i,j}$. Encode disequality predicates with negations, e.g., encode $i \neq j$ with $\neg e_{i,j}$
2. Recover the lost transitivity of equality by conjoining ϕ_{enc} with explicit transitivity constraints jointly denoted by ϕ_{trans}. ϕ_{trans} is a formula over ϕ_{enc} 's variables and, possibly, auxiliary variables.

- The Boolean formula $\phi_{enc} \wedge \phi_{trans}$ should be satisfiable if only if ϕ^E is satisfiable. Further, it should be possible to construct a satisfying assignment to ϕ^E from an assignment to the $e_{i,j}$ variables.

Decision Procedure for Equality Logic

- We will first investigate methods that solve Equality Logic. Uninterpreted functions are eliminated with one of the reduction schemes.

- Our starting point: the E-Graph $G^E(\phi^E)$

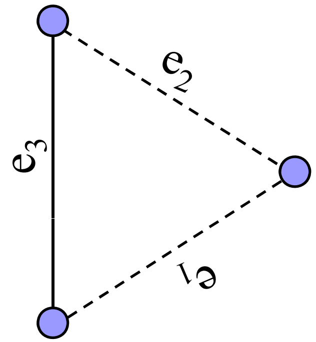
- Recall: $G^E(\phi^E)$ represents an abstraction of $\phi^{E:}$
  It represents ALL equality formulas with the same set of equality predicates as ϕ^E

From Equality to Propositional Logic

Bryant & Velev 2000: the *Sparse* method

$$\phi^E = x_1 = x_2 \wedge x_2 = x_3 \wedge x_1 \neq x_3$$

$$\phi_{enc} = e_1 \wedge e_2 \wedge \neg e_3$$

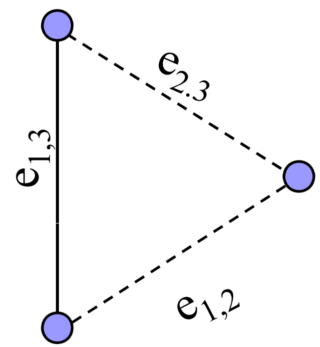


- Encode all edges with Boolean variables
  - (note: for now, ignore polarity)
  - This is an abstraction
  - Transitivity of equality is lost!
  - Must add transitivity constraints!

From Equality to Propositional Logic

$$\phi^E = x_1 = x_2 \land x_2 = x_3 \land x_1 \neq x_3$$

$$\phi_{enc} = e_{1,2} \land e_{2,3} \land \neg e_{1,3}$$



- For each cycle add a transitivity constraint

$$\phi_{trans} = (e_{1,2} \land e_{2,3} \rightarrow e_{1,3}) \land (e_{1,2} \land e_{1,3} \rightarrow e_{2,3}) \land (e_{1,3} \land e_{2,3} \rightarrow e_{1,2})$$

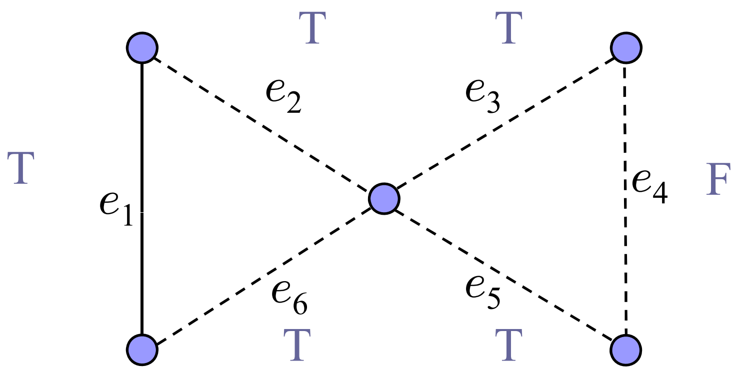
Check: $\phi_{enc} \land \phi_{trans}$

Transitivity constraints

- $\phi^E = x_1 = x_2 \wedge ((x_2 = x_3 \wedge x_1 \neq x_3) \vee (x_1 \neq x_2))$
- $\phi_{enc} = e_{1,2} \wedge ((e_{2,3} \wedge \neg e_{1,3}) \vee \neg e_{1,2})$
- ϕ^E is satisfiable, then ϕ_{enc} is satisfiable .
  - Not vice versa
    - ϕ_{enc} is satisfiable, but not ϕ^E
    - We need transitivity constraints ϕ_{trans} !
- For variables x_i, x_j, and x_i, the constraint $e_{i,j} \wedge e_{j,k} \rightarrow e_{i,k}$ is called a transitivity constraint
  - Transitivity constraints can be added to T for every three variables in ϕ^E (although it is possible to fine a small subset of them that is still sufficient)

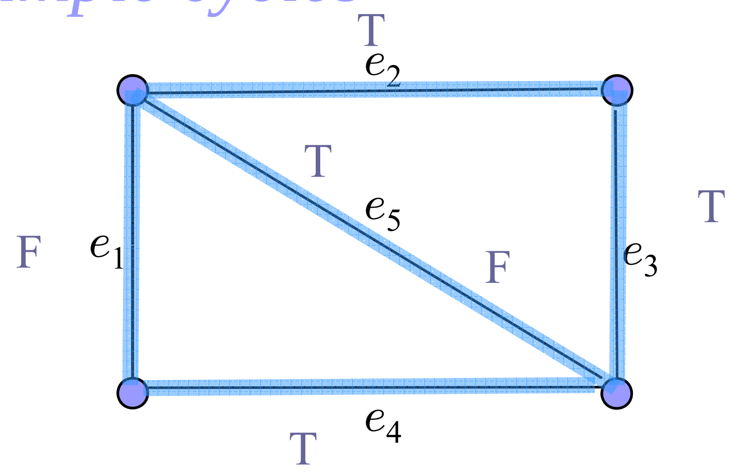
From Equality to Propositional Logic

- There can be an exponential number of cycles, so let's try to make it better.
- Thm: *it is sufficient to constrain simple cycles only*



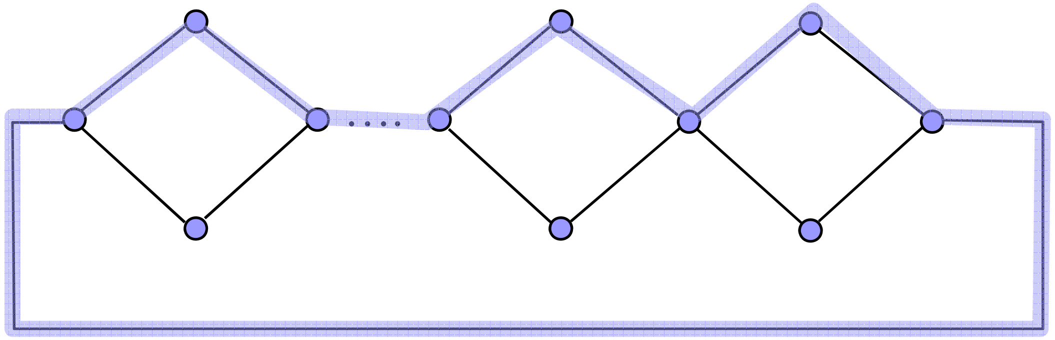
From Equality to Propositional Logic

- Still, there is an exponential number of simple cycles.
- Def. A chord of a cycle is an edge connecting two non-adjacent nodes of the cycle
- Thm [Bryant & Velev]: *It is sufficient to constrain chord-free simple cycles*



From Equality to Propositional Logic

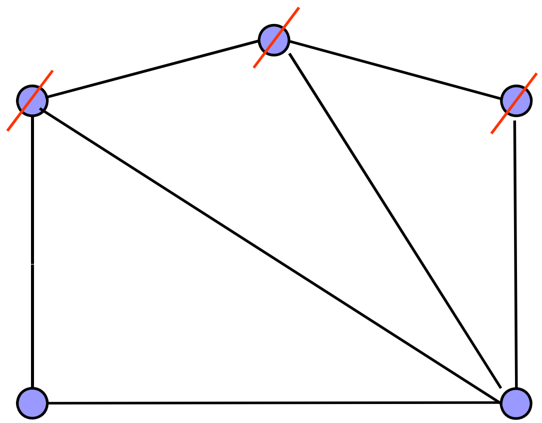
- Still, there can be an exponential number of chord-free simple cycles…



- Solution: make the graph ‘chordal’ by adding edges.

From Equality to Propositional Logic

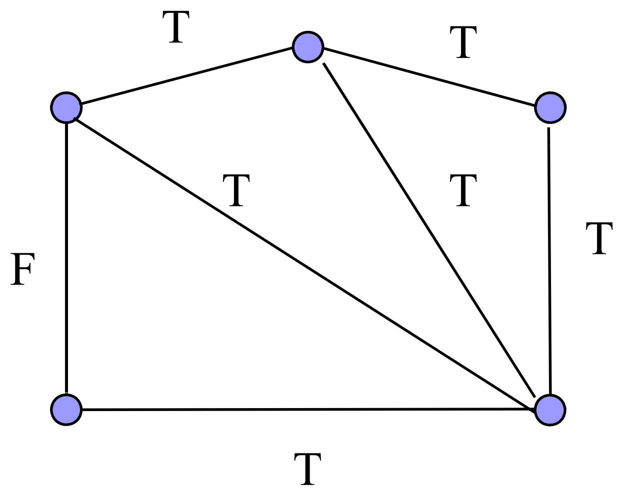
- Dfn: A graph is chordal iff every cycle of size 4 or more has a chord.
- How to make a graph chordal ? eliminate vertices one at a time, and connect their neighbors.



From Equality to Propositional Logic

- Once the graph is chordal, we can constrain only the triangles.

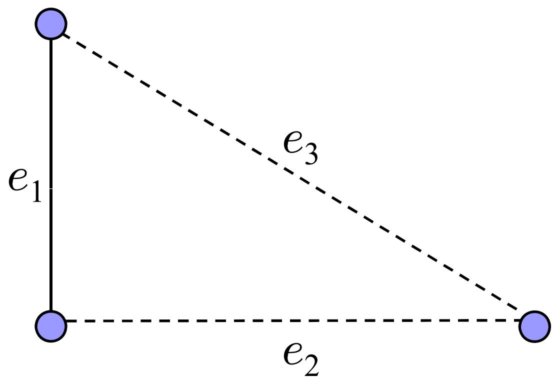
Contradiction!



- Note that this procedure adds not more than a polynomial # of edges, and results in a polynomial no. of constraints.

Further Improvement

- So far we did not consider the polarity of the edges.
- Claim: in the following graph $\phi_{trans} = e_3 \wedge e_2 \rightarrow e_1$ is sufficient, since contradictory cycles can be constrained more efficiently with polarity information



- See "Generating minimum transitivity constraints in P-time for deciding equality logic" in Satisfiability Modulo Theories (SMT) 2007