DDDLIB: A Library For Solving Quantified Difference Inequalities

18th Conference on Automated Deduction
Copenhagen, Denmark, July 27-30, 2002

Jesper Blak Møller
[jm@it.edu]

Department of Innovation
The IT University of Copenhagen
[www.it.edu/people/jm]

Outline of Talk

1. Difference Inequalities
2. Timed Guarded Commands
3. Difference Decision Diagrams
4. DDDLIB

Difference Inequalities

- A first-order logic over difference inequalities:

$$\phi \ ::= \ x - y \leq d \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \exists x.\phi$$

- No absolute constraints. Can be introduced using a variable z representing “zero”.
- No Boolean variables. Can be modelled by $x - x' \leq 0$.
- We want to solve the decision problem:

TSAT: is ϕ satisfiable?

- TSAT is **PSPACE**-complete [Koubarakis 94].

Timed Guarded Commands

- The logic naturally induces a **timed guarded command language**:

$$\phi \rightarrow x := \mathbf{any}\ x'.\ \phi$$

- Can be used for modelling dynamic systems with continuous time. Advancing time:

$$z := \mathbf{any}\ z'.\ (z' \leq z \wedge P_{\mathrm{post}})$$

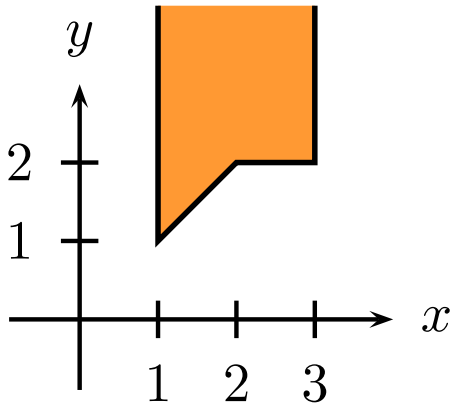
- Some of the variables are *clocks*, others model discrete state through a Boolean encoding.
- A decision procedure for the logic makes it possible to verify such systems.

Difference Decision Diagrams: Introduction

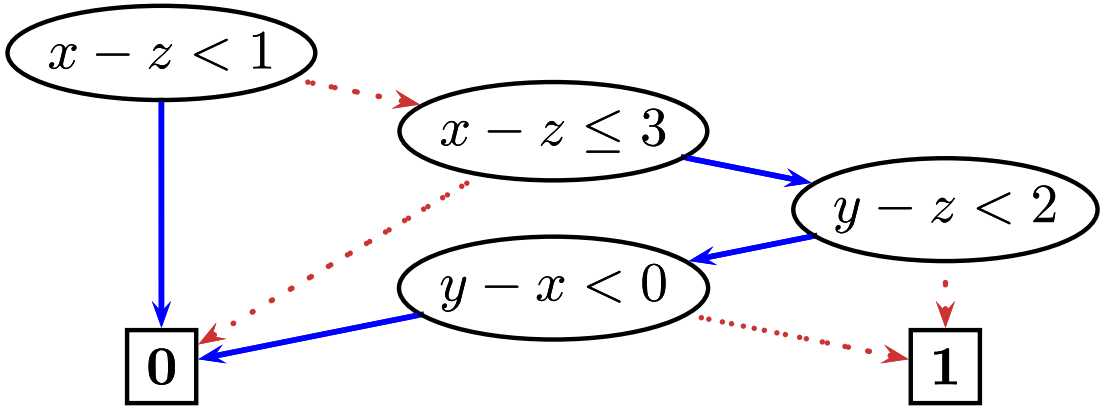
- Implicit representation of the logic :

$$\phi \ ::= \ x - y \leq d \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \exists x.\phi \, .$$

- $\phi \ = \ 1 \leq x - z \leq 3 \ \wedge \ (y - z \geq 2 \ \vee \ y - x \geq 0)$



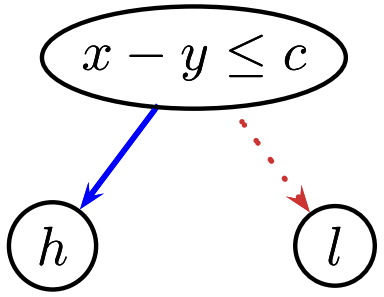
(x, y)-plot for $z = 0$



Difference decision diagram

DDDs: Local Reduction Rules

- A **difference decision diagram** is a directed, acyclic graph with two types of nodes:

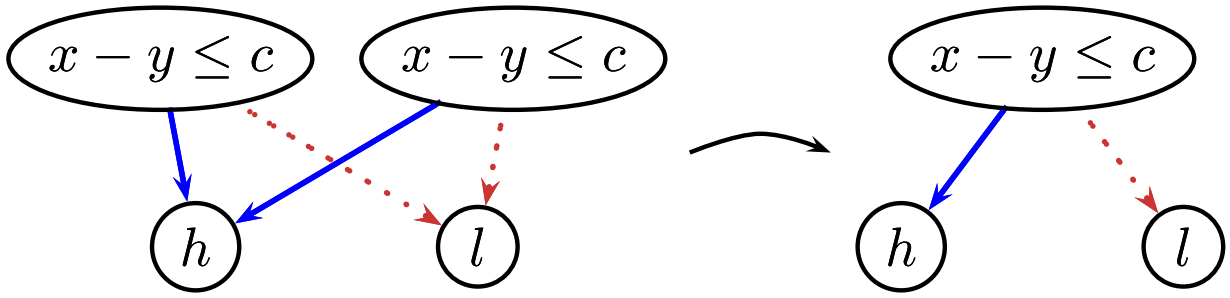


if $x - y \leq c$ **then** h **else** l

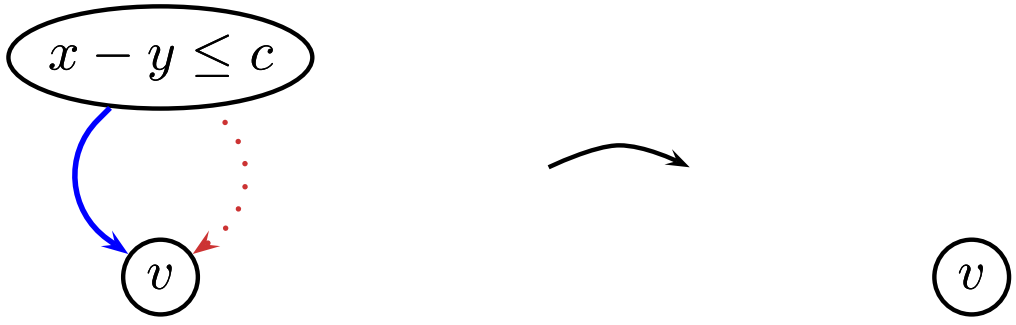
$\boxed{0}$ $\boxed{1}$ **false** respectively **true**

DDDs: Local Reduction Rules (2)

- No duplicate vertices:

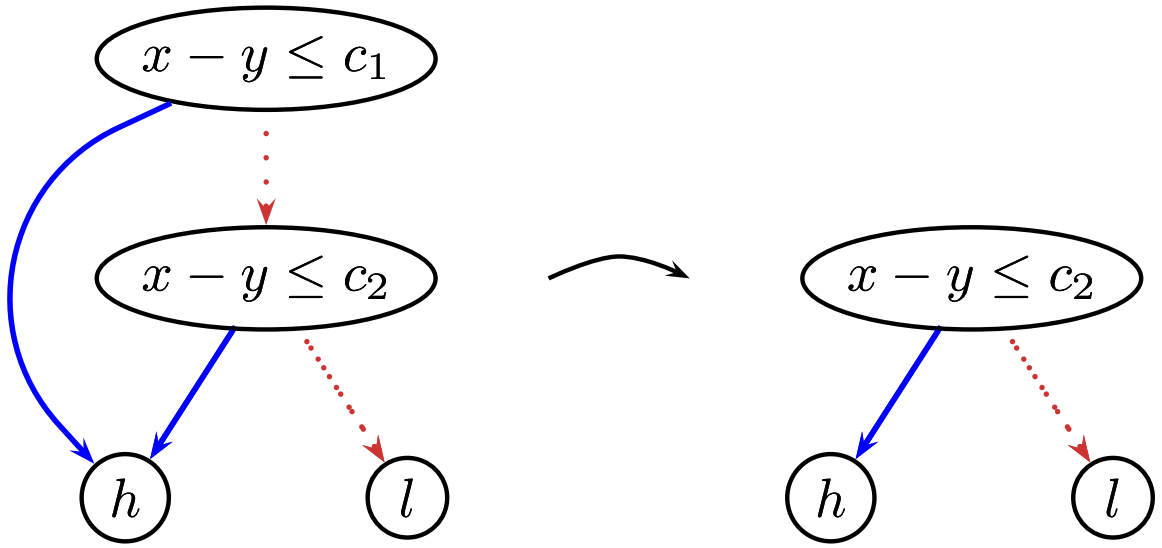


- Different high- and low-branches:



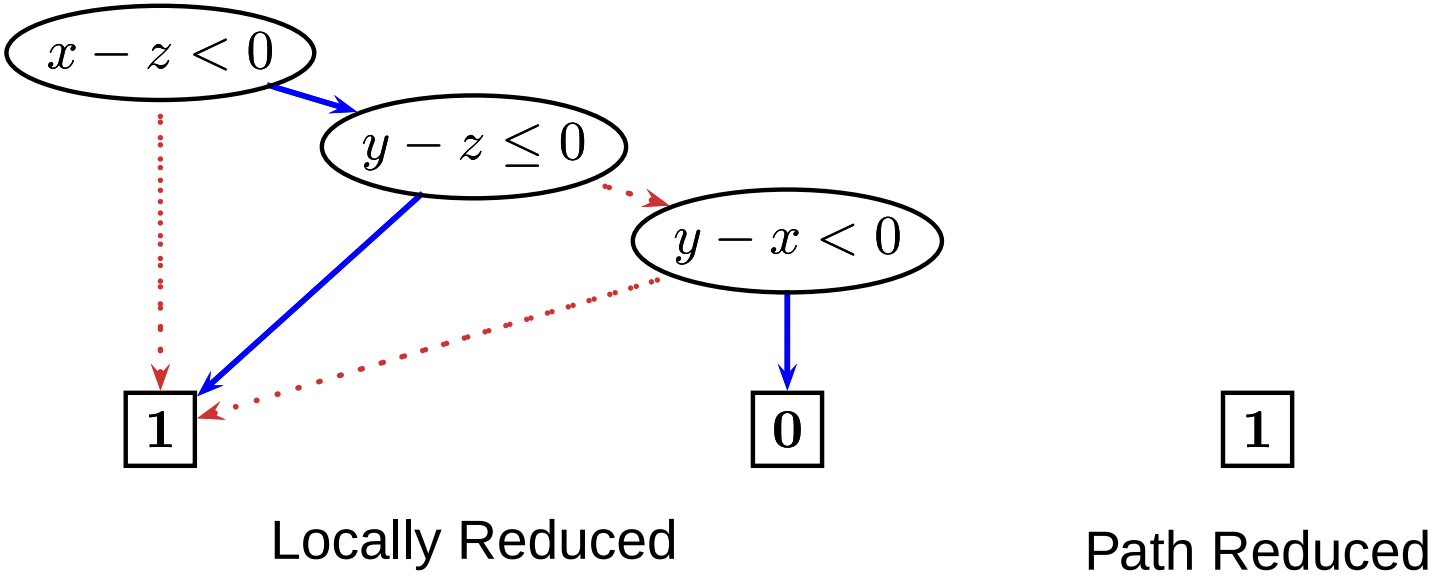
DDDs: Local Reduction Rules (3)

- No redundant tests:



DDDs: Path Reduction

- A **path-reduced DDD** is a locally reduced DDD where all paths are feasible
- For example:



- **Theorem:** A path-reduced DDD is a tautology if and only if it is the terminal vertex 1.

DDDs: Construction & Manipulation

- DDD for $x - y \leq c$: *mk*$(x, y, c, 1, 0)$
- DDD for $\phi_1 \wedge \phi_2$: *apply*$(\wedge, u_1, u_2)$
  using a generalization of the BDD *apply* algorithm
- DDD for $\neg\phi$: *apply*$(\neg_1, u, \_)$
- DDD for $\exists x.\phi$: *exists*(x, u)
  using Fourier–Motzkin quantifier elimination
- DDD is a tautology: *path_reduce*$(u) = \mathbf{1}$
  using Bellman–Ford's algorithm

DDDLIB: An Implementation of DDDs

- **DDDLIB:** Kernel written in C with high-level interfaces for C++ and Standard ML
- Objective Caml interface available [Sorea 01a]
- DDDs have been integrated in a verification tool for **infinite state systems** [Abdulla & Nylén 00]
- DDDs are part of SRIs **TEMPO model checker** for event-recording automata [Sorea 01b]
- DDDs are used in the **CALCULEMUS project** for bounded model checking [Cimatti et al.]
- Available from [www.it.edu/research/ddd] for non-commercial use