Leveraging Linear Programming for pseudo-Boolean solving

Jo Devriendt †, Jan Elffers †, Ambros Gleixner ‡, Jakob Nordström †
† KTH Royal Institute of Technology, Sweden
‡ Zuse Institut Berlin, Germany
jhmde@kth.se
Three abbreviations

- CP = constraint programming
- PB = pseudo-Boolean
- LP = linear programming
CP demo!

- Pigeonhole problem in IDP
  - http://dtai.cs.kuleuven.be/krr/idp-ide/?src=c01635bf2172be67577f0856684fb3f8
  - Timeout on small problem sizes
Pigeonhole problem in IDP
- http://dtai.cs.kuleuven.be/krr/idp-ide/?src=c01635bf2172be67577f0856684fb3f8
  - Timeout on small problem sizes

Viewed as an integer linear program, specification is *rationally infeasible*
- polynomially decidable!
CP demo!

- Pigeonhole problem in IDP
  - http://dtai.cs.kuleuven.be/krr/idp-ide/?src=c01635bf2172be67577f0856684fb3f8
    - Timeout on small problem sizes
- Viewed as an integer linear program, specification is *rationally infeasible*
  - polynomially decidable!
- Why is IDP's performance this bad?
Explanation: *resolution* is bad at pigeonhole

- IDP uses MiniSatID as backend CP solver
- MiniSatID uses *lazy clause generation* algorithm
Explanation: resolution is bad at pigeonhole

- IDP uses MiniSatID as backend CP solver
- MiniSatID uses lazy clause generation algorithm
  - explains propagations through clauses
Explanation: *resolution* is bad at pigeonhole

- IDP uses MiniSatID as backend CP solver
- MiniSatID uses *lazy clause generation* algorithm
  - explains propagations through clauses
  - learns clause from conflict (*no-good*)
Explanation: *resolution* is bad at pigeonhole

- IDP uses MiniSatID as backend CP solver
- MiniSatID uses *lazy clause generation* algorithm
  - *explains* propagations through clauses
  - *learns* clause from conflict (*no-good*)
  - builds *resolution proofs*
Explanation: 
resolution is bad at pigeonhole

- IDP uses MiniSatID as backend CP solver
- MiniSatID uses lazy clause generation algorithm
  - explains propagations through clauses
  - learns clause from conflict (*no-good*)
  - builds resolution proofs
- Resolution is infamously bad at pigeonhole [1]
Potential solution: use *cutting-planes* proof system

- Generalization of resolution
- Great for rationally infeasible problems [2]
  - e.g. pigeonhole
Potential solution: use *cutting-planes* proof system

- Generalization of resolution
- Great for rationally infeasible problems [2]
  - e.g. pigeonhole
- Used by many *pseudo-Boolean* solvers
  - decide feasibility of 0-1 integer linear programs
  - e.g. RoundingSat, Sat4J
  - learn *linear inequality* from conflict
Potential solution:
use cutting-planes proof system

- Generalization of resolution
- Great for rationally infeasible problems [2]
  - e.g. pigeonhole
- Used by many pseudo-Boolean solvers
  - decide feasibility of 0-1 integer linear programs
  - e.g. RoundingSat, Sat4J
  - learn linear inequality from conflict
- RoundingSat has no problem with pigeonhole 😊
Potential solution: use *cutting-planes* proof system

- Generalization of resolution
- Great for rationally infeasible problems [2]
  - e.g. pigeonhole
- Used by many *pseudo-Boolean* solvers
  - decide feasibility of 0-1 integer linear programs
  - e.g. RoundingSat, Sat4J
  - learn *linear inequality* from conflict
- RoundingSat has no problem with pigeonhole

- RoundingSat fails on several other rationally infeasible problems
Summary so far

1. CP and PB solvers struggle on rational infeasibility
2. Stronger underlying proof system helps on some, but not all problems
3. Rational feasibility is polynomially decidable [3]
Summary so far

1. CP and PB solvers struggle on rational infeasibility
2. Stronger underlying proof system helps on some, but not all problems
3. Rational feasibility is polynomially decidable [3]

How to exploit rational infeasibility during search?
Modern search loop

Propagation
Modern search loop

Propagation ➔ Conflict?
Modern search loop

Propagation → Decide unassigned variable

Conflict? → no
Modern search loop

Propagation → Decide unassigned variable

Conflict? → no

Conflict? → yes → Learn constraint
Modern search loop

1. Propagation
2. Decide unassigned variable
   - no
   - yes
     - Conflict?
     - yes
     - Learn constraint
     - no
     - Backjump
Modern search loop

with rational feasibility oracle

- Decide unassigned variable
  - Rational infeasibility?
    - Query oracle
      - no
        - Conflict?
          - yes
            - Learn constraint
              - Backjump
          - no
            - Propagation
Modern search loop

with rational feasibility oracle

- Decide unassigned variable
  - no
  - Rational infeasibility?
    - no
    - Query oracle
      - no
      - Conflict?
        - yes
        - Learn constraint
        - Backjump
      - yes
      - Conflict?
        - yes
        - Learn constraint
      - no
      - Backjump
Modern search loop
with rational feasibility oracle

Decide unassigned variable
no
Rational infeasibility?
Query oracle
no
Conflict?
yes
Learn constraint
Backjump

Propagation
Modern search loop

with rational feasibility oracle

- Decide unassigned variable
  - no
  - Rational infeasibility?
    - no
    - Query oracle
      - yes
      - Learn constraint
        - Backjump
    - Conflict?
      - yes
      - Learn constraint
        - Backjump
  - yes

Linear Programming (LP) solvers

- **In:**
  - conjunction of linear constraints
  - variable bounds
  - objective function

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax \leq b \\
& \quad x \geq 0
\end{align*}
\]
Linear Programming (LP) solvers

• In:
  - conjunction of linear constraints
  - variable bounds
  - objective function

• Out: either
  - SAT: optimal rational solution
  - UNSAT: Farkas multipliers
    - defines violated linear combination of input constraints

\[
\text{minimize } c^T x \\
\text{subject to } \begin{align*}
A x & \leq b \\
x & \geq 0
\end{align*}
\]
Linear Programming (LP) solvers

- **In:**
  - conjunction of linear constraints
  - variable bounds
  - objective function

- **Out:** either
  - SAT: optimal rational solution
  - UNSAT: Farkas multipliers
    - defines violated linear combination of input constraints

\[
\begin{aligned}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad A x \leq b \\
& \quad x \geq 0
\end{aligned}
\]
Modern search loop

with LP solver call

Decide unassigned variable

no

Rational infeasibility?

yes

Extract Farkas multipliers

no

Query LP solver *

no

Conflict?

yes

Learn (Farkas) constraint

Backjump

Propagation
Two technical problems

- LP solvers are relatively slow
  - Limit calls to LP solver
  - Limit LP solver running time
Two technical problems

- LP solvers are relatively slow
  - Limit calls to LP solver
  - Limit LP solver running time
- LP solvers use inexact floating point arithmetic
  - Independently calculate Farkas constraint with exact multiple precision
  - Verify falsifiedness of Farkas constraint
Working implementation with PB solver

- Trivial conversion between PB and LP constraints
Working implementation with PB solver

- Trivial conversion between PB and LP constraints
- PB solver RoundingSat
Working implementation with PB solver

- Trivial conversion between PB and LP constraints
- PB solver RoundingSat
- LP solver SoPlex
Experiments!

- 5 solver configurations
  - RoundingSat
  - **RoundingSat+SoPlex**
  - SCIP
  - Sat4J
  - Sat4J-CP
- 3000s on 16GiB machines
- 4 benchmark families:
  - PB12
  - PB16
  - MIPLIB
  - PROOF
Experiments!

PB12

- RoundingSat
- RoundingSat + SoPlex
- SCIP
- Sat4J
- Sat4J-CP

Number of solved instances vs. Timeout limit (s)
Experiments!

PB16

- RoundingSat
- RoundingSat + SoPlex
- SCIP
- Sat4J
- Sat4J-CP

Number of solved instances

Timeout limit (s)
Experiments!

MIPLIB

Number of solved instances

_TIMEOUT LIMIT (s)_

- RoundingSat
- RoundingSat + SoPlex
- SCIP
- Sat4J
- Sat4J-CP
Experiments!

PROOF

- RoundingSat
- RoundingSat + SoPlex
- SCIP
- Sat4J
- Sat4J-CP

Number of solved instances vs Timeout limit (s)
Experiments indicate

- RoundingSat+SoPlex ≥ RoundingSat
  - small LP overhead at worst, huge speedups at best
Experiments indicate

- RoundingSat+SoPlex ≥ RoundingSat
  - small LP overhead at worst, huge speedups at best
- Only on MIPLIB, SCIP > RoundingSat+SoPlex
Experiments indicate

- RoundingSat+SoPlex $\geq$ RoundingSat
  - small LP overhead at worst, huge speedups at best
- Only on MIPLIB, SCIP > RoundingSat+SoPlex
- SoPlex does not like PB12
Future work

- Add learnt constraints to LP solver *
Future work

- Add learnt constraints to LP solver *
- Use LP cuts as learnt constraints
Future work

- Add learnt constraints to LP solver *
- Use LP cuts as learnt constraints
- Exploit rational solutions to constraints
Future work

- Add learnt constraints to LP solver *
- Use LP cuts as learnt constraints
- Exploit rational solutions to constraints
- Improve LP solving performance on PB benchmarks
Future work

• Add learnt constraints to LP solver *
• Use LP cuts as learnt constraints
• Exploit rational solutions to constraints
• Improve LP solving performance on PB benchmarks
• Optimization!
How about the CP setting?
How about the CP setting?

- *Linearizations* of CP models exist [4]
How about the CP setting?

- Linearizations of CP models exist [4]
- Farkas constraint can be *rounded* to clausal no-good
How about the CP setting?

- Linearizations of CP models exist [4]
- Farkas constraint can be *rounded* to clausal no-good
- No theoretical obstacles for our approach to work for lazy clause generation CP
How about the CP setting?

- Linearizations of CP models exist [4]
- Farkas constraint can be rounded to clausal no-good
- No theoretical obstacles for our approach to work for lazy clause generation CP

Questions to NordConsNet:
- would LP integration be helpful for CP solvers?
- does any CP solver do this already?
Conclusion

- Use LP solver to tackle on-the-fly rational infeasibility
Conclusion

- Use LP solver to tackle on-the-fly rational infeasibility
- Implemented sound integration of LP solver in PB solver
Conclusion

- Use LP solver to tackle on-the-fly rational infeasibility
- Implemented sound integration of LP solver in PB solver
- Experiments indicate small LP overhead at worst, huge speedups at best
Conclusion

- Use LP solver to tackle on-the-fly rational infeasibility
- Implemented sound integration of LP solver in PB solver
- Experiments indicate small LP overhead at worst, huge speedups at best
- In theory, technique can be lifted to CP
Conclusion

- Use LP solver to tackle on-the-fly rational infeasibility
- Implemented sound integration of LP solver in PB solver
- Experiments indicate small LP overhead at worst, huge speedups at best
- In theory, technique can be lifted to CP

Thanks for your attention!

Questions?
References

[1] The intractability of resolution - 1985 - Haken