Parameterized Complexity in Constraint Programming

Toby Walsh
NICTA and UNSW

Newcastle, March 2010
Parameterized Complexity in Constraint Programming

Mostly based on [Bessiere, Hebrard, Hnich, Kiziltan, Quimper, Walsh, AAAI-08]
Parameterized Complexity
in
Constraint Programming

End of talk includes work with George Katsirelos and Nina Narodytska
Motivation

Beyond the NP-hardness of CP
- Parameterized complexity

Some common themes
- Tree width
- Cutsets & backdoors
- Dynamic programming
Constraint Programming 101

- Variables
  - Each with a finite domain of values

- Constraints
  - Allowed tuples of values
Constraint Programming 101

- Variables
  - $X_{ij} \in \{1, \ldots, 9\}$

- Constraints
  - $\text{AllDiff}(X_{11}, \ldots, X_{19}), \ldots$
  - $\text{AllDiff}(X_{11}, \ldots, X_{33}), \ldots$
  - $\text{AllDiff}(X_{11}, \ldots, X_{91}), \ldots$
  - $\text{AllDiff}(X_{11}, \ldots, X_{33}), \ldots$
Constraint Programming 101

- Backtracking search
  - Try $X_{11}=1$
  - Else $X_{11}=2$
  - ..

- Propagation
  - If $X_{11}=1$ then $X_{12} \neq 1$
  - If
Constraint Programming 101

• Global constraints
  – Capture common patterns
  – Efficient propagation algorithms
  – E.g. AllDiff
Parameterized complexity

• Useful insight into (in)tractability of CP
• Some common themes
  – Tree width
  – Backdoors/cutsets
  – Dynamic programming
Parameterized complexity

- Useful insight into (in)tractability of CP
- Some common themes
  - Tree width
  - Backdoors/cutsets
  - Dynamic programming

Practical algorithms?
Parameterized complexity

• Useful insight into (in)tractability of CP
• Some common themes
  – Tree width
  – Backdoors/cutsets
  – Dynamic programming

Practical algorithms?
Parameterized complexity

- Useful insight into (in)tractability of CP
- Some common themes
  - Tree width ☺
  - Backdoors/cutsets ☹
  - Dynamic programming

Practical algorithms?
Parameterized complexity

• Useful insight into (in)tractability of CP
• Some common themes
  – Tree width ☺
  – Backdoors/cutsets ☹
  – Dynamic programming ☺

Practical algorithms?
Parameterized complexity & CP

Complexity **of**
searching for a solution

*Exponential sized*
search tree

Complexity **within** the search for a solution

*Propagation at each node of this tree*
Complexity of search

- Finding solutions
  - Or proving unsatisfiability
- Consider the whole search tree!
k-consistency

- Arc-consistency (k=2)
  - If we assign any value to any variable, we can find a satisfying assignment for any other variable
  - E.g. $X > Y$, $\text{dom}(X) = \text{dom}(Y) = \{1,2\}$ then enforcing AC sets $X=2$, $Y=1$
k-consistency

- Arc-consistency (k=2)
  - If we assign any value to any variable, we can find a satisfying assignment for any other variable
  - Takes $O(ed^2)$ time to enforce
  - If constraint graph is acyclic then enforcing AC, we can find solutions backtrack free
k-consistency

- **k-consistency** ($k \geq 2$)
  - If we assign any values to any $k-1$ variables, we can find a satisfying assignment for any other variable
  - Takes $O(d^k)$ time to enforce
  - If constraint graph has tree width $\leq k$ then enforcing $k$-consistency, we can find solutions backtrack free \[\text{[Freuder 82]}\]
Complexity within search

- Global constraints are often on limits of tractability
- At each node of search tree
  - Propagation may be very costly
**NValues(X1,..,Xn,N)**

- Values often represent resources
  - E.g. frequency assignment
- NValues satisfied iff
  - \(|\{X_i \mid 1 \leq i \leq n\}| = N\)
  - Generalization of AllDifferent
NValues(X1,..,Xn,N)

- Propagation is NP-hard
  - Finding a satisfying assignment
  - Reduction from 3SAT
  - $X_i \in \{z_i, -z_i\}$ for $1 \leq i \leq n$
  - $X_{n+j} \in \{z_a, -z_b, z_c\}$ if jth clause is $z_a$ or $-z_b$ or $z_c$
  - $N=n$
NValues(X1,..,Xn,N)

• Consider two different parameters
  – $k = |\bigcup \text{dom}(X_i)|$ then fixed parameter tractable
  – $k = \max(\text{dom}(N))$ then $W[2]$-hard
NValues(X₁,..,Xₙ,N)

• $k = \left| \bigcup \text{dom}(X_i) \right|

  - Define automaton that accepts only those sequences $X₁,..,Xₙ,N$ that satisfy NValues
  - States of automaton are sets of values used so far
  - Global REGULAR constraint propagates this in $O(ndQ) = O(nd2^k)$ time using dynamic programming
\begin{itemize}
    \item $k = \max(\text{dom}(N))$ then \textit{W[2]}-hard
        \begin{itemize}
            \item Hitting set is smallest set that \textit{hits} every set in a collection
            \item Hitting set is \textit{W[2]}-complete in size of hitting set
            \item Immediate reduction
        \end{itemize}
\end{itemize}
Backdoors

- **Strong backdoor**
  - Subset of variables which give a polynomial subproblem
  - Correlated with problem hardness [Williams, Gomes, Selman IJCAI-03]
    [Kilby, Slaney, Thiebaux, Walsh, AAAI-05]
Backdoors

- Many of our *fixed-parameter tractability* results for global constraints exploit
  - *Cycle cutsets* that are *backdoors* into an acyclic (and thus polynomial) subproblem
  - Once cycle cutset is instantiated, we can use 2-consistency (aka arc-consistency) to solve problem
Disjoint([X1,..,Xn],[Y1,..,Ym])

- Xi≠Yj for any i, j
- Useful in scheduling and time-tabling problems
- NP-hard to propagate
  - Simple reduction (homework exercise :-)

Newcastle, March 2010
Disjoint([X1,..,Xn],[Y1,..,Ym])

• Fixed-parameter tractable in
  \[ k = |\bigcup \text{dom}(X_i) \cap \bigcup \text{dom}(Y_j)| \]

• Try all \(2^k\) possible subsets for
  \[ \bigcup X_i \cap \bigcup \text{dom}(Y_j) \]
  – Each is a cycle cutset
  – Remaining \(X_i\) and \(Y_j\) are disjoint

Newcastle, March 2010
Disjoint([X1,..,Xn],[Y1,..,Ym])

• Fixed-parameter tractable in

\[ k = |\bigcup \text{dom}(X_i) \cap \bigcup \text{dom}(Y_j)| \]
Disjoint([X1,..,Xn],[Y1,..,Ym])

• Fixed-parameter tractable in
  \[ k = |\bigcup \text{dom}(X_i) \cap \bigcup \text{dom}(Y_j)| \]
  \[ k = |\bigcup \text{dom}(X_i) \cup \bigcup \text{dom}(Y_j)| \]
Other FPT results

• **Uses**([X1,..,Xn],[Y1,..,Ym])
  – $\bigcup \text{dom}(X_i) \subseteq \bigcup \text{dom}(Y_j)$
  – $k = |\bigcup \text{dom}(Y_j)|$

• **Among**([X1,..,Xn],S,N)
  – $N = |\{i \mid X_i \in S\}|$
  – $k = |\text{ub}(S) \setminus \text{lb}(S)|$
Other FPT results

- **ROOTS**([X_1,..,X_n],S,T)
  - S={i | X_i \in T}
  - Used to encode Among, AtMost, AtLeast, Uses, Domain, ...
  - k=|ub(T) \setminus lb(T)|

- **CardPath**([X_1,..,X_n],C,N)
  - N = |\{i | C(X_i,..,X_i+p)\}|
  - Used to encode Regular, Sequence, Contiguity, Lex
FPT results in symmetry

- Lex Leader
  - General symmetry breaking method
- Double Lex
  - Specialized method for row and column symmetry
Symmetry

• All interval series
  – $X_1,\ldots, X_{11} = 3, 7, 4, 6, 5, 0, 10, 1, 9, 2, 8$
  – Problem from musical composition
Symmetry

• All interval series
  – $X_1, \ldots, X_{11} = 3, 7, 4, 6, 5, 0, 10, 1, 9, 2, 8$
  – Value symmetry: $i \mapsto 10 - i$
  – $X_1, \ldots, X_{11} = 7, 3, 6, 4, 5, 10, 0, 9, 1, 8, 2$
Symmetry

• All interval series
  – $X_1,..,X_{11} = 3,7,4,6,5,0,10,1,9,2,8$
  – Variable symmetry: $X_i \leftrightarrow X_{12-i}$
  – $X_1,..,X_{11} = 8,2,9,1,10,0,5,6,4,7,3$
Symmetry constraints

• Eliminate symmetric solutions
  – If we want such solutions, easy to generate
  – This reduces search exponentially in best case
  • Both symmetric
Symmetry constraints

- All interval series
- Value symmetry
  - $X_1,..,X_{11} \leq_{lex} 10-X_1,..,10-X_{11}$
- Variable symmetry
  - $X_1,..,X_{11} \leq_{lex} X_{11},..,X_1$
Lex Leader

- General method for any group of symmetries $\Sigma$
  - Leave just the smallest (in a lex ordering) in each symmetry class
  - $X_1, \ldots, X_n \preceq_{\text{lex}} \sigma(X_1), \ldots, \sigma(X_n)$ for all $\sigma \in \Sigma$
Lex Leader

• General method for any group of symmetries $\Sigma$
  - $X_1,..,X_n \leq_{\text{lex}} \sigma(X_1),..,\sigma(X_n)$ for all $\sigma \in \Sigma$
  - E.g. inversion value symmetry that maps $a$ onto $d-a$
  - $X_1,..,X_n \leq_{\text{lex}} d-X_1,..,d-X_n$
Lex Leader

- General method for any group of symmetries $\Sigma$
  - Let $\text{LexLeader}(\Sigma, X_1, \ldots, X_n)$ hold iff $X_1, \ldots, X_n \leq_{\text{lex}} \sigma(X_1), \ldots, \sigma(X_n)$ for all $\sigma \in \Sigma$
  - Propagating $\text{LexLeader}$ is NP-hard
Lex Leader

• General method for any group of symmetries \( \Sigma \)
  – Let \( \text{LexLeader}(\Sigma, X_1, \ldots, X_n) \) hold iff \( X_1, \ldots, X_n \leq_{\text{lex}} \sigma(X_1), \ldots, \sigma(X_n) \) for all \( \sigma \in \Sigma \)
  – Propagating \( \text{LexLeader} \) is NP-hard
  – Fixed-parameter tractable in \( k=|\Sigma| \)
Lex Leader

- General method for any group of symmetries $\Sigma$
  - Let $\text{LexLeader}(\Sigma, X_1, \ldots, X_n)$ hold iff $X_1, \ldots, X_n \leq_{\text{lex}} \sigma(X_1), \ldots, \sigma(X_n)$ for all $\sigma \in \Sigma$
  - Propagating LexLeader is NP-hard
  - Fixed-parameter tractable in $k = |\Sigma|$
- Actually number of generators of $\Sigma$!
Lex Leader

• General method for any group of symmetries $\Sigma$
  – Let $\text{LexLeader}(\Sigma,X_1,..,X_n)$ hold iff $X_1,..,X_n \leq_{\text{lex}} \sigma(X_1),..,\sigma(X_n)$ for all $\sigma \in \Sigma$
  – Define automaton that accepts only sequences $X_1$ to $X_n$ that satisfy lex inequalities
  – States are which subset of inequalities have been satisfied so far, $2^k$ such states
Row & Col Symmetry

- Matrix model
  - Arrays of decision variables

- Row and col symmetry
  - Interchangeable rows & columns

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\end{bmatrix}
\]
Double Lex

- DoubleLex
  - Lex ordering rows (breaks all row symmetry)
  - Lex ordering cols (breaks all col symmetry)
  - But leaves some row/col symmetries
Double Lex

- **DoubleLex**
  - Lex order rows
  - Lex order cols

- Eliminates most symmetry
  - NP-hard to check if all symmetry is eliminated

Newcastle, March 2010
Double Lex

- DoubleLex
  - Lex order rows
  - Lex order cols
- Propagating DoubleLex is NP-hard
Double Lex

- DoubleLex
  - Lex order rows
  - Lex order cols

- Propagating DoubleLex is NP-hard
  - Took 10 years to show this!
Double Lex

- DoubleLex
  - Lex order rows
    - 001
  - Lex order cols
    - 010 and 011
- Does not eliminate all row & col symmetry
  - Look in the shadow!
Row & Col Symmetry

• Breaking all row & col symmetry
  – NP-hard in general
  – Fixed parameter tractable in $k = \min(\#\text{rows}, \#\text{cols})$
  – Simple observation:

*All row symmetry is eliminated by lex ordering rows*

*Consider all $2^k$ possible permutations of columns where $k = \#\text{cols}$*
Conclusions

• Parameterized complexity gives useful new insight into
  – Complexity of search in CP
  – Complexity of propagation in CP
Conclusions

• Parameterized complexity gives useful new insight into
  – Complexity of search in CP
  – Complexity of propagation in CP

• Some common themes
  – Tree width
  – Cycle cutsets & backdoors
  – Dynamic programming & automata