Decomposing Global Constraints into SAT

Christian Bessiere, George Katsirelos, Nina Naroditskaya,
Claude-Guy Quimper, **Toby Walsh**

Decomposing Global Constraints into SAT

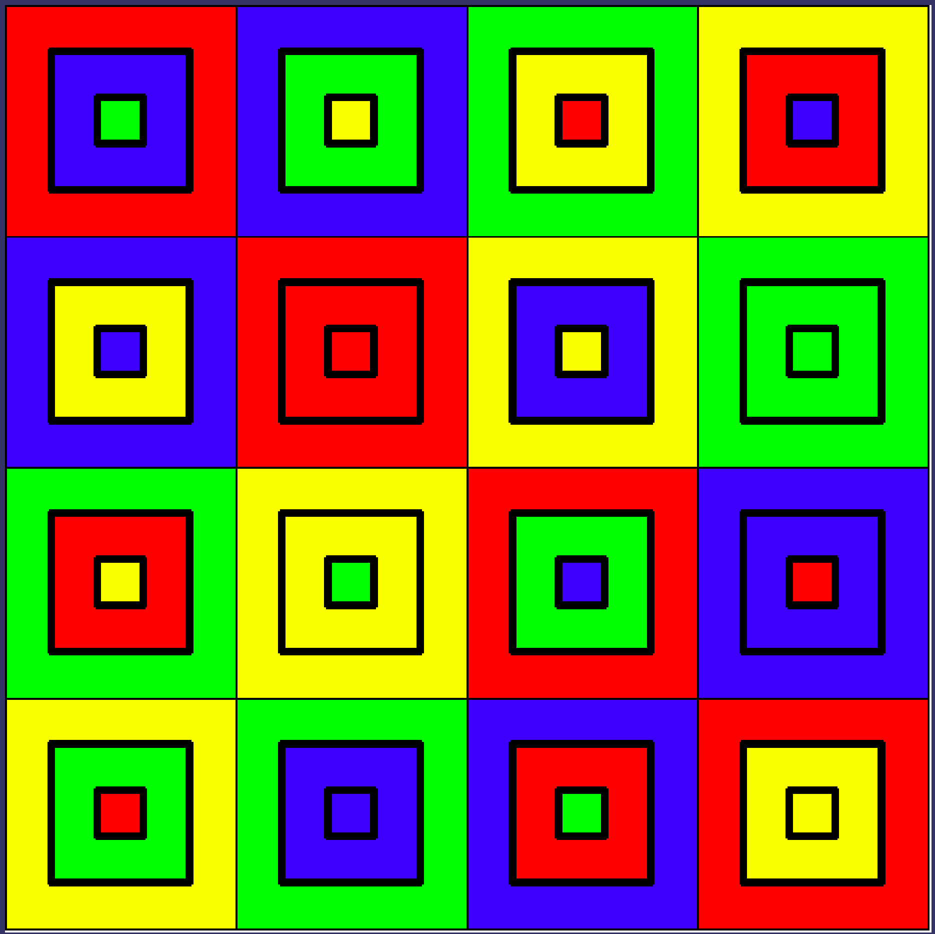
(alias: why we need constraint programming!)

Christian Bessiere, George Katsirelos, Nina Naroditskaya,
Claude-Guy Quimper, **Toby Walsh**

Motivation

- Mature and effective technologies in constraint programming
  - Global constraints
  - Search methods
  - Branching heuristics
  - …
- Many successful applications
  - Scheduling, configuration, …

Global constraints



- Capture common patterns
  - E.g. permutation = AllDifferent
- Associated propagator
  - Efficient inference methods

Motivation

- Many global constraints can be effectively decomposed
  - REGULAR constraint in a call centre rostering problem [Quimper & Walsh CP06]
  - REGULAR, SEQUENCE, TABLE constraints [Bacchus CP06]
  - GRAMMAR constraint [Quimper & Walsh CP07]
  - ALL DIFFERENT constraint [Bessiere et al IJCAI 09]

Why not just decompose all global constraints?

- MIP decomposes everything into linear inequalities after all
- Many free advantages
  - nogood learning, incremental propagators, impact based heuristics, …

What can you do in SAT and what can you only do in CP?

Decomposing GLOBALs

- Successful examples
  - REGULAR
  - SEQUENCE
  - CFG
  - ALL DIFFERENT
  - NVALUES
  - GCC
  - ...

Decomposing GLOBALs

- Successful examples
  - **REGULAR**
  - SEQUENCE
  - CFG
  - **ALL DIFFERENT**
  - NVALUES
  - GCC
  - ...

Main results

- Some surprisingly complex propagation algorithms for global constraints can be efficiently decomposed
- However, there exist other propagation algorithms which have an exponential size decomposition

REGULAR constraint

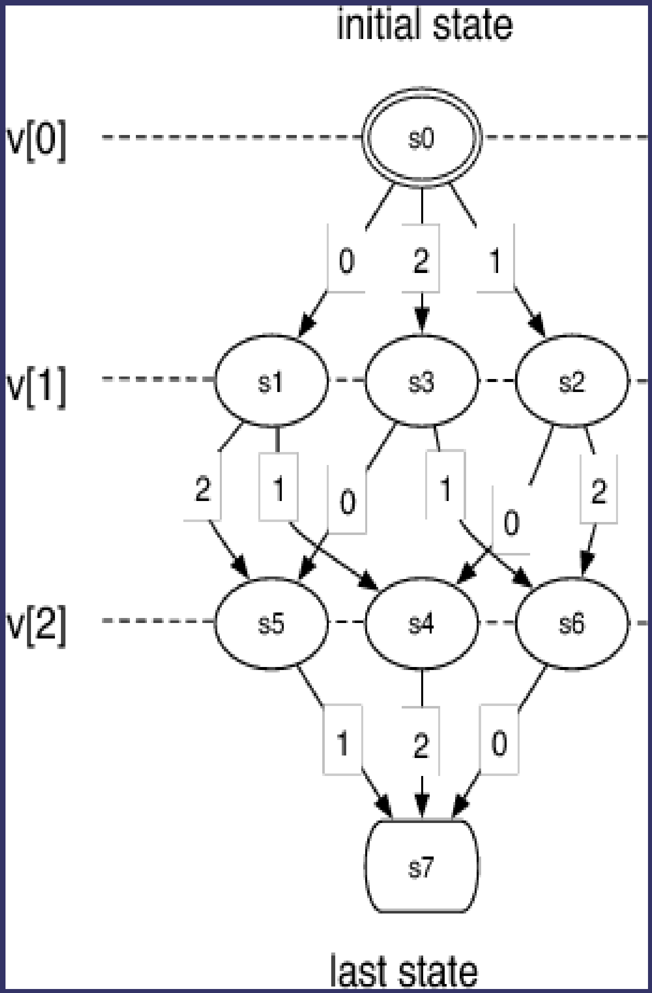
- Sequence of vars defines a regular language
- Useful in rostering and many other problems

Shift = Days | Nights | Days Shift | Nights Shift

- Days = d | d Days
- Nights = n | n n | n n n

REGULAR constraint

- Sequence of vars defines a regular language
- Complex propagator based on dynamic programming
  - Enforces domain consistency in O(ndQ)



REGULAR constraint

- Simple decomposition
  - Does not hinder propagation
  - Again in O(ndQ) time and space
- Introduce sequence of vars for state of automaton after jth step
  - Then simply post transition constraints
  - Qi+1 = t(Qi,Xi)

REGULAR constraint

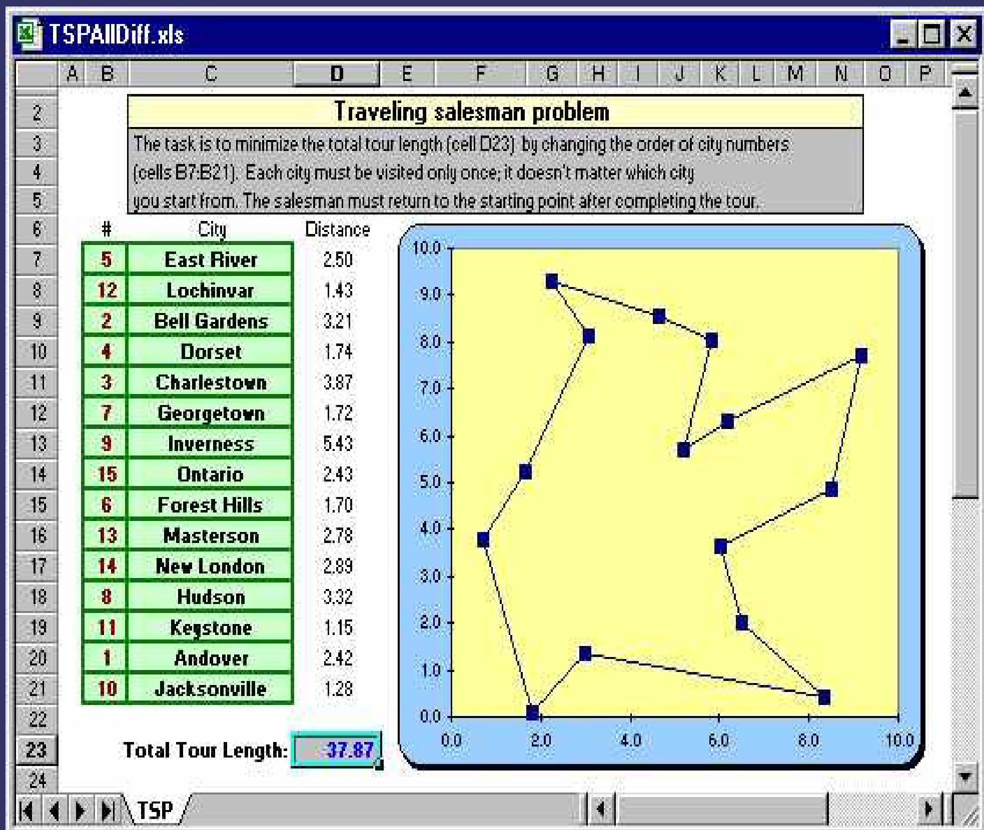
- Rostering example

Shift = Days | Nights | Days Shift | Nights Shift

  - Days = 0 | 0 Days
  - Nights = 1 | 1 1 | 1 1 1

- Transition constraints
  - $Q_{i+1} = X_i * (Q_i + X_i)$
  - Q_i in [0,3]

AllDifferent constraint



- ➲ One of the oldest global constraints
  - Appeared in ALICE [Laurier 78]
- ➲ One of the most useful global constraints
  - Timetabling
  - Scheduling
  - Routing
  - …

AllDifferent constraint

- AllDifferent(X1,..,Xn) iff

$Xi \neq Xj$ for i<j

E.g AllDifferent(1,2,4) but not
AllDifferent(1,2,1)

AllDifferent constraint

- Decompose into $O(n^2)$ inequalities
  - This hinders propagation
  - $X1, X2, X3 \in \{1,2\}$
  - $X1 \neq X2$, $X1 \neq X3$, $X2 \neq X3$ are domain consistent
  - However, AllDifferent(X1,X2,X3) has no solution

*Can we decompose whilst maintaining a **global** view?*

AllDifferent constraint

- Range/bound consistency propagator can be encoded into SAT
  - Based on Hall intervals
- Domain consistency propagator requires exponential size SAT encoding
  - Based on circuit complexity

AllDifferent constraint

- Bound consistency
  - Max and min in each domain have bound support
  - Bound support = satisfying assignment in which each var is bet max and min
- Range consistency
  - Every value in each domain has bound support
- Domain consistency
  - Every value in each domain has support
  - Support = satisfying assignment in which each var is assigned value from its domain

AllDifferent constraint

- Bound and range consistency
  - Central concept is **Hall interval**
  - Hall interval = interval of *m* values which contains domains of *m* variables
    - Other variables must find their bound supports outside of this!

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2	*	*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2	*	*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[1,1] is a Hall interval

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2	*	*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[1,1] is a Hall interval, X5 consumes interval

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2	*	*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[1,1] is a Hall interval, X2 $\neq$ 1

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2		*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[1,1] is a Hall interval, X2 $\neq$ 1

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2		*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[3,4] is a Hall interval, contains X1 and X3

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2		*	*	*	
X3			*	*	
X4		*	*	*	*
X5	*				

[3,4] is a Hall interval, X2∩[3,4]=0, X4∩ [3,4]=0

AllDifferent constraint

	1	2	3	4	5
X1			*	*	
X2		*			
X3			*	*	
X4		*			*
X5	*				

[3,4] is a Hall interval, $X2 \cap [3,4] = 0$, $X4 \cap [3,4] = 0$

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$

Can be given directly to MIP solver

Easily encoded into SAT

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$
- Does not hinder propagation
  - Domain consistency on decomposition achieves range consistency on AllDifferent

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$
- Does not hinder propagation
  - Domain consistency on decomposition achieves range consistency on AllDifferent
  - Time complexity $O(nd^3)$ down branch

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$
- Does not hinder propagation
  - Bound consistency on decomposition achieves bound consistency on AllDifferent

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$
- Does not hinder propagation
  - Bound consistency on decomposition achieves bound consistency on AllDifferent
  - Time complexity $O(nd^2)$ down branch

AllDifferent constraint

- Simple decomposition
  - $A_{ilu}=1$ iff $Xi \in [l,u]$
  - $\sum A_{ilu} \leq u-l+1$
- Does not hinder propagation
  - Unit propagation on SAT encoding achieves range consistency on AllDifferent

Other decompositions

- Similar interval based decompositions for many other global constraints:
  - NValues
  - GCC
  - Common
  - Used
  - …

That shows what we can do in SAT …

Now for what we cannot do in SAT

Main result

Polynomial size CNF decomposition of constraint propagator

iff

Polynomial size monotone circuit

Proof outline

Constraint propagator

$\Leftrightarrow$

Constraint checker

CNF Decomposition

$\Leftrightarrow$

Monotone circuit

All reductions polynomia

Constraint propagator

- Function f:$\mathcal{P}$(D)$\Rightarrow$$\mathcal{P}$(D)
  - Monotone
    - I.e. smaller input, smaller output
  - Contracting
    - I.e. output $\subseteq$ input
  - Idempotent
    - I.e. f(f(input)) = f(input)
  - Correct
    - Only prunes values that have no support
    - When domain wipes out, there is no support

Constraint checker

- Function $f:\mathcal{P}(D) \Rightarrow \{0,1\}$
  - Monotone
    - I.e. smaller input, smaller output
  - Correct
    - I.e. if f(input)=0 then there is no support

Polynomial time constraint propagator

iff

Polynomial time constraint checker

CNF decomposition of constraint checker

- Input vars of CNF encode characteristic function of D(X)
  - Can also have auxiliary vars
- One output var
  - Set to false by unit propagation iff constraint checker returns 0

Monotone circuits

- DAG of ANDs and ORs
- Computes exactly the monotone Boolean functions
  - Note: our result has nothing to say about P=NP. There are, for instance, poly time monotone Boolean functions which require superpoly size monotone circuit

Main result

Polynomial size CNF decomposition of constraint propagator/checker

iff

Polynomial size monotone circuit

Circuit complexity

- We can now use lower bounds on monotone circuits
  - E.g. Domain consistency propagator for AllDifferent computes perfect matching in bipartite value graph
  - Smallest monotone circuit to compute such a matching is super-polynomial

Circuit complexity

- We can now use lower bounds on monotone circuits
  - Thus, we conclude that the smallest CNF encoding of a domain consistency propagator for AllDifferent is super-polynomial in size

Circuit complexity

- We can now use lower bounds on monotone circuits
  - Thus, we conclude that the smallest CNF encoding of a domain consistency propagator for AllDifferent is super-polynomial in size

Unfortunately there aren't too many monotone circuit complexity results to exploit

Possible (but unlikely) escapes

- Don’t represent domains explicitly but just bounds
  - Exponentially more compact
- Use monotone arithmetic circuits

Conclusions

- What we can do in SAT
  - ***Some*** global constraints
  - E.g. Domain consistency on REGULAR, and range/bound consistency on AllDifferent

- What we cannot do in SAT
  - ***All*** global constraints
  - E.g. Domain consistency on AllDifferent