The Extended Global Cardinality Constraint: An Empirical Survey

Journal Track, published in Artificial Intelligence, 2011

Peter Nightingale
University of St Andrews

Global Cardinality Constraint

- Introduced by J.-C. Régin (1996)
- Has become one of the key global constraints in CP
- EGCC(X, V, C)
  - X is a vector of target variables
  - V is a vector of domain values of interest
  - C is a vector of cardinality variables
- For each value V_i with cardinality variable Ci, there are C_i occurrences of V_i in X in any solution
  - Other values are free

Example

- Car Sequencing problem
- We need five cars of type A, two cars of type B, one of type C, two of type D
- EGCC(Seq, [A,B,C,D], [5,2,1,2])
- One solution for this constraint:

Seq=	A	B	A	C	A	B	A	D	A	D

- Car Sequencing also has sub-sequence constraints
  - These can also be expressed with EGCC – models A and AB in experiments

Motivation

- Paper is partly empirical survey of existing algorithms....
  - Quimper’s algorithm vs Régin’s algorithm
  - Three algorithms for cardinality variables
  - Many more
- ... And partly new optimisations for EGCC
  - Dynamic partitioning
  - Dynamic triggers

Motivation

- Help future solver implementors
  - Simple algorithms better than complex ones
    - Despite big-O complexity
  - Insight into which parts of code to optimise
    - Despite big-O complexity, again
  - How to prune cardinality variables
- Techniques for EGCC might apply elsewhere
  - Dynamic partitioning for graph/network constraints

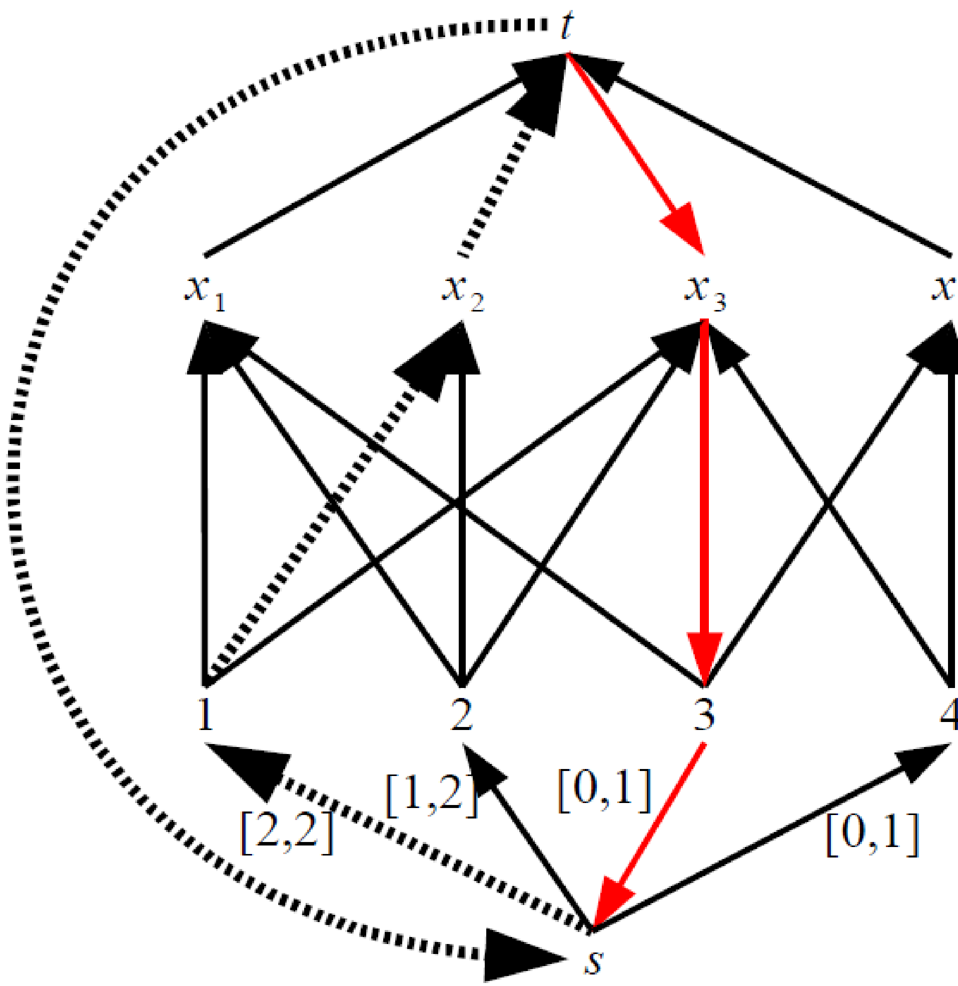
Pruning the Target Variables: Background

Sketch of Régin's algorithm:

Phase 1

Find a maximal (integral) flow in a network representing the EGCC constraint

The maximal flow corresponds to a satisfying assignment of the target variables



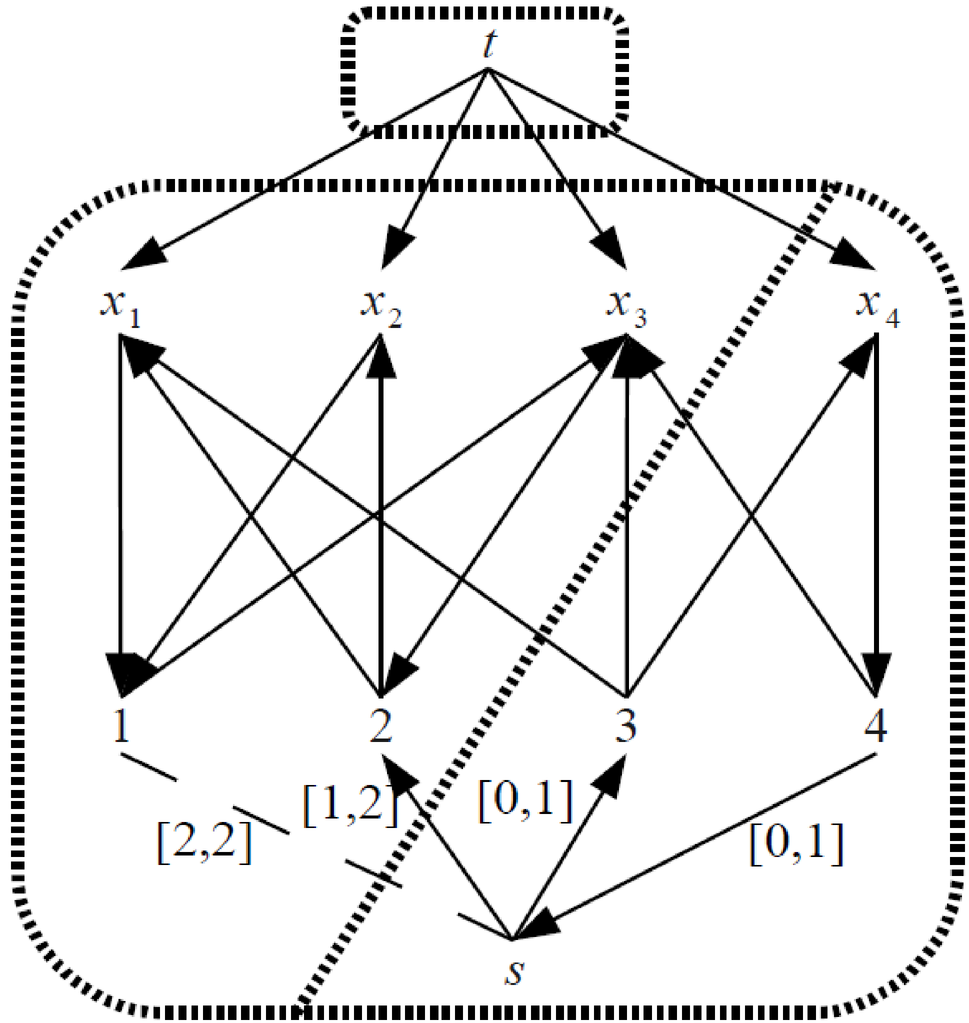
Augmenting path for edge $(s,1)$

Pruning the Target Variables: Background

Sketch of Régin's algorithm:

Phase 1

We have a maximal flow, edges in the flow are reversed.



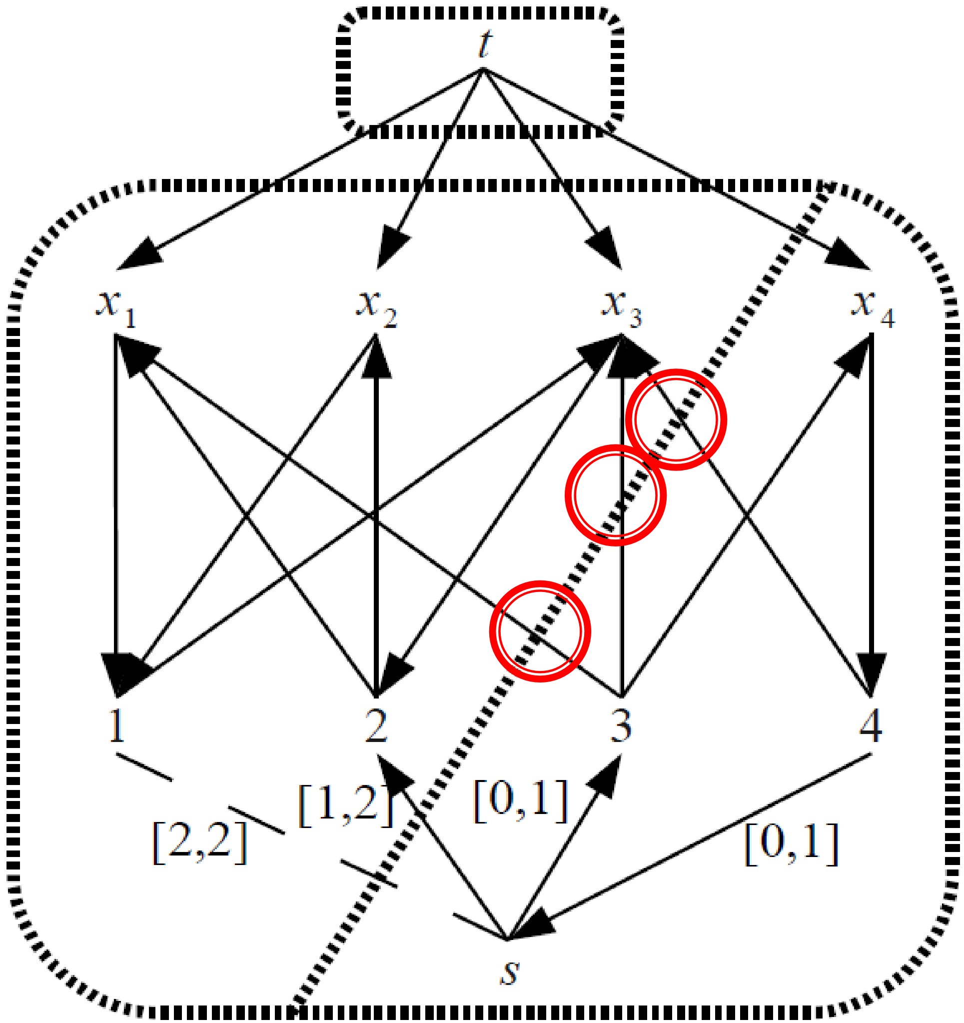
SCCs of the residual graph

Pruning the Target Variables: Background

Phase 2

Compute the Strongly Connected Components (SCCs)

Value->Variable edges crossing from one SCC to another must be pruned

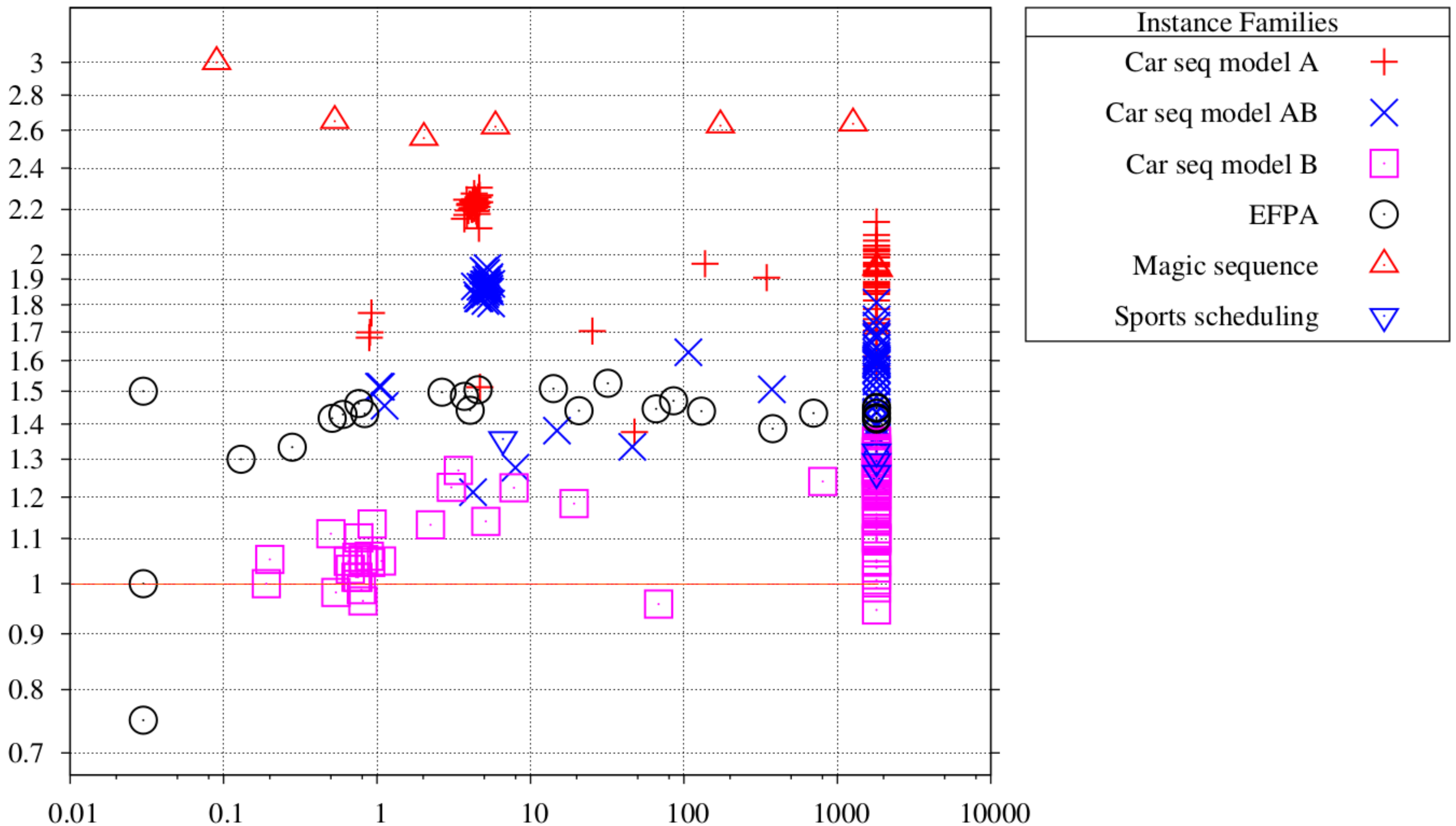


SCCs of the residual graph

Pruning the Target Variables

- Two algorithms
- Régin (1996)
  - Finds one maximal flow, SCC analysis once
  - Network flow, $O(n^2d)$
- Quimper et al (2004)
  - Divides the EGCC into two constraints for the lower and upper bounds (on cardinality)
  - Finds two matchings and runs SCC analysis twice
  - Bipartite matching, $O(n^{1.5}d)$

Pruning the Target Variables



Same search, comparing node rate
Régin's algorithm much more efficient

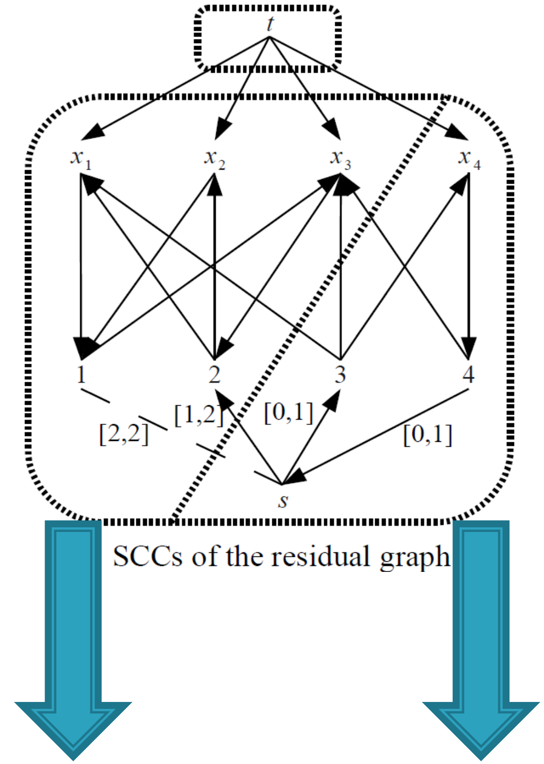
Pruning the Target Variables

- Why is Régin's algorithm faster?

1. First phase of both algorithms dominate the big-O analyses
   - However first phase is incremental and in practice very quick
   - Second phase (SCC analysis) takes most of the time
   - First phase less than 15% in profiles
   - Quimper's algorithm does SCC analysis twice!
2. Simple BFS flow algorithm faster than Hopcroft Karp or similar on 'small' problems (see AllDifferent)

Dynamic Partitioning

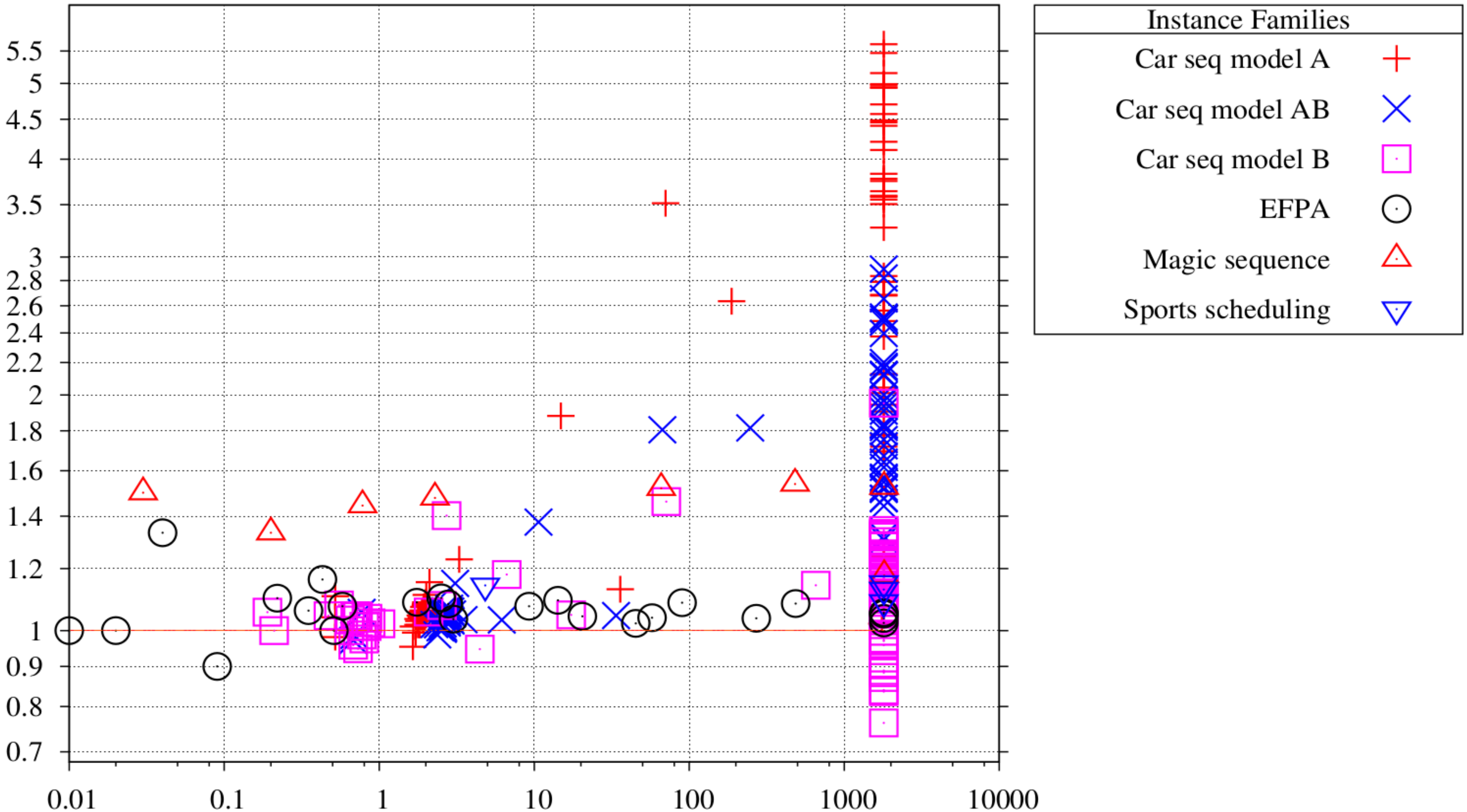
- When the network splits into multiple SCCs, partition the constraint
- Changes to variables only trigger the relevant cells
- Changes SCC analysis from Θ(nd) to O(nd)
- Makes SCC incremental



x_1,x_2,x_3 1,2

x_4 3,4

Dynamic Partitioning



Very useful optimization

Dynamic Partitioning

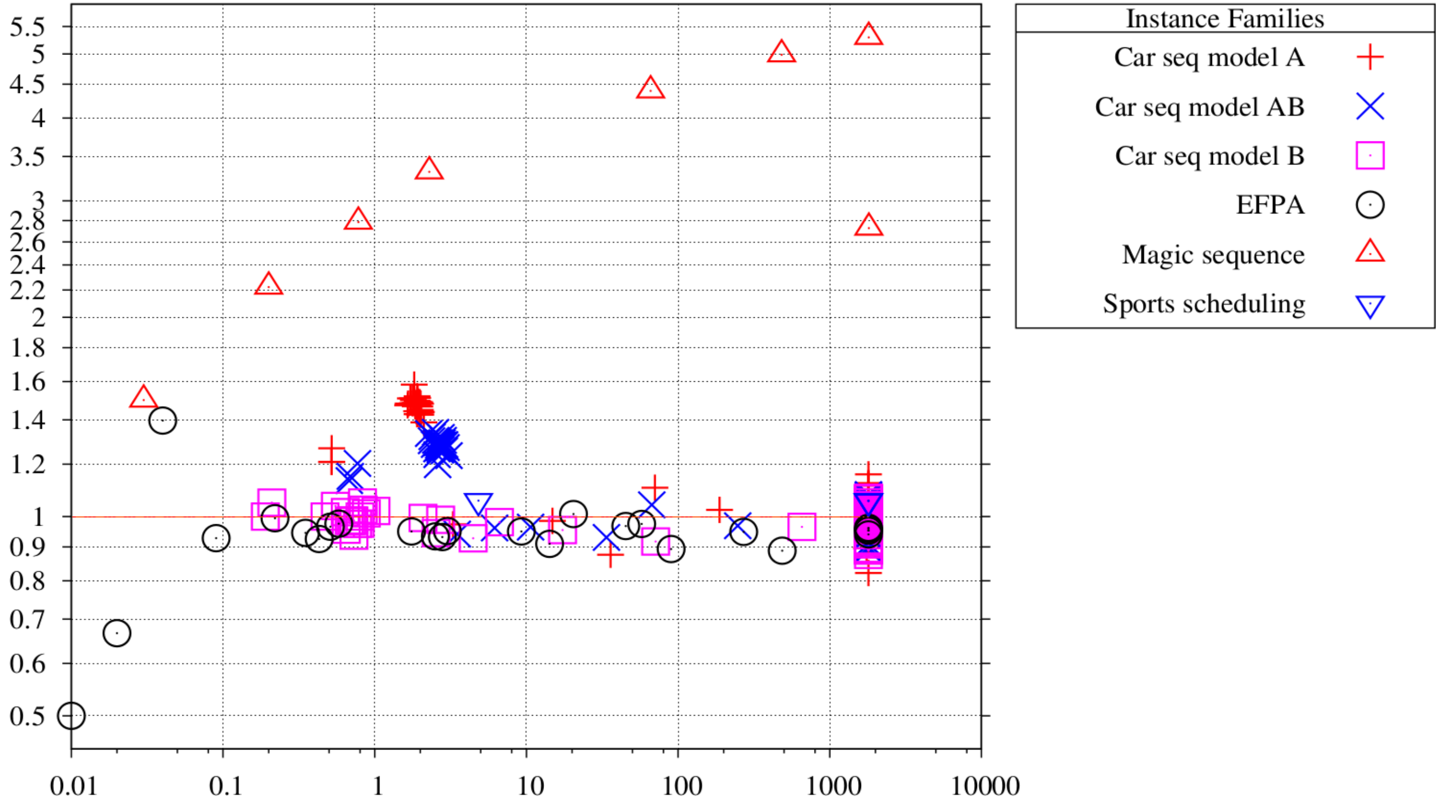
- Works well for AllDifferent and EGCC
- Promising for other graph- or network-based constraints, e.g.
  - Multiset-Same or Same-With-Cardinalities
  - Graph connected constraint – partitions at the bridge edges
  - If the underlying graph or network partitions, split the constraint

Dynamic Triggers

- Katriel identified important values of target variables
- If a value is not important, it will not cause any propagation
- Approx 3n values (n target variables)
- Retrieve approximation of the important value set from SCC analysis – very cheap
- When EGCC triggered, check if any important values removed

Dynamic Triggers

- Doesn't help much – except *magic sequence* with a very unusual structure
- Still triggers for *some* value of each variable
- Might be valuable with very large domains of target variables

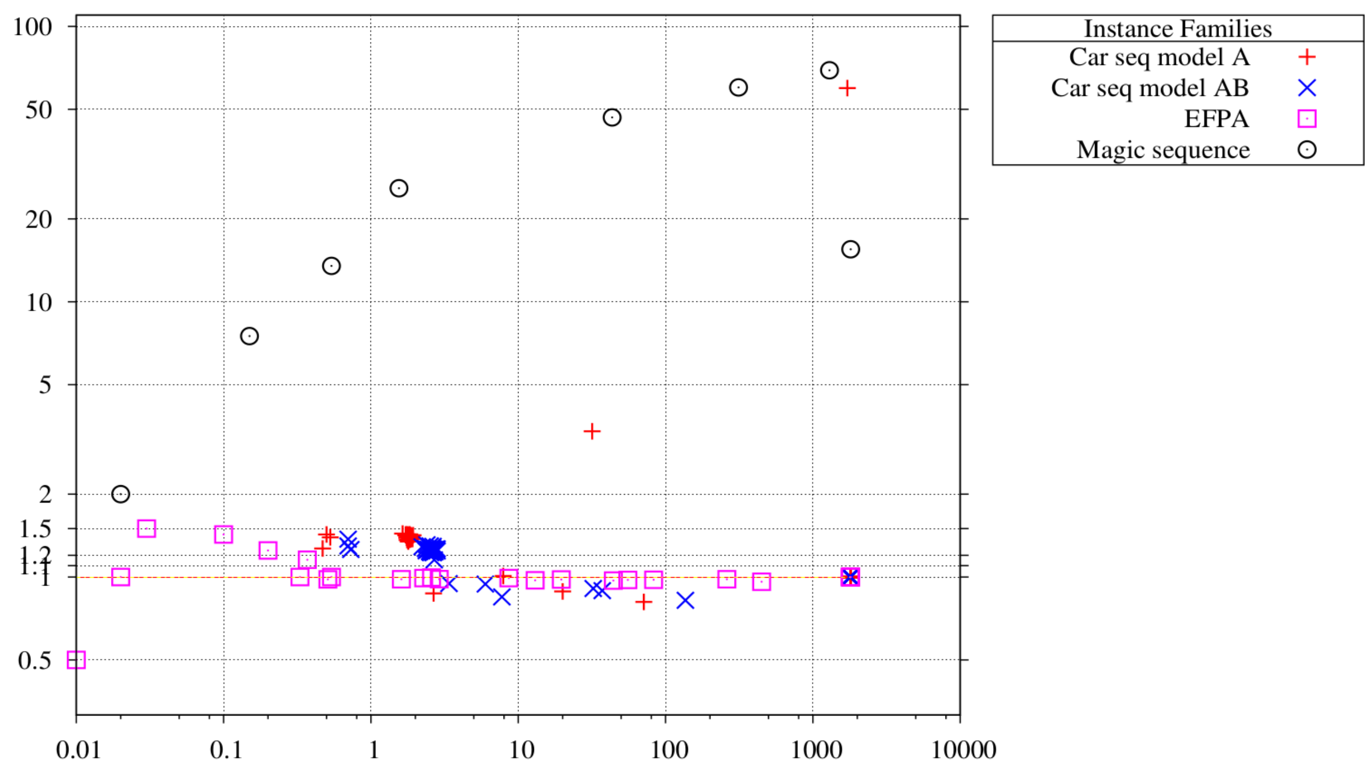


Pruning the Cardinality Variables

- Simple – for each value:
  - Count occurrences in the domains of the target variables (upper bound)
  - Count target variables assigned to the value (lower bound)
- Sum – simple plus implied sum constraint
  - Cardinality variables sum to number of target variables
  - Only correct when all values are listed

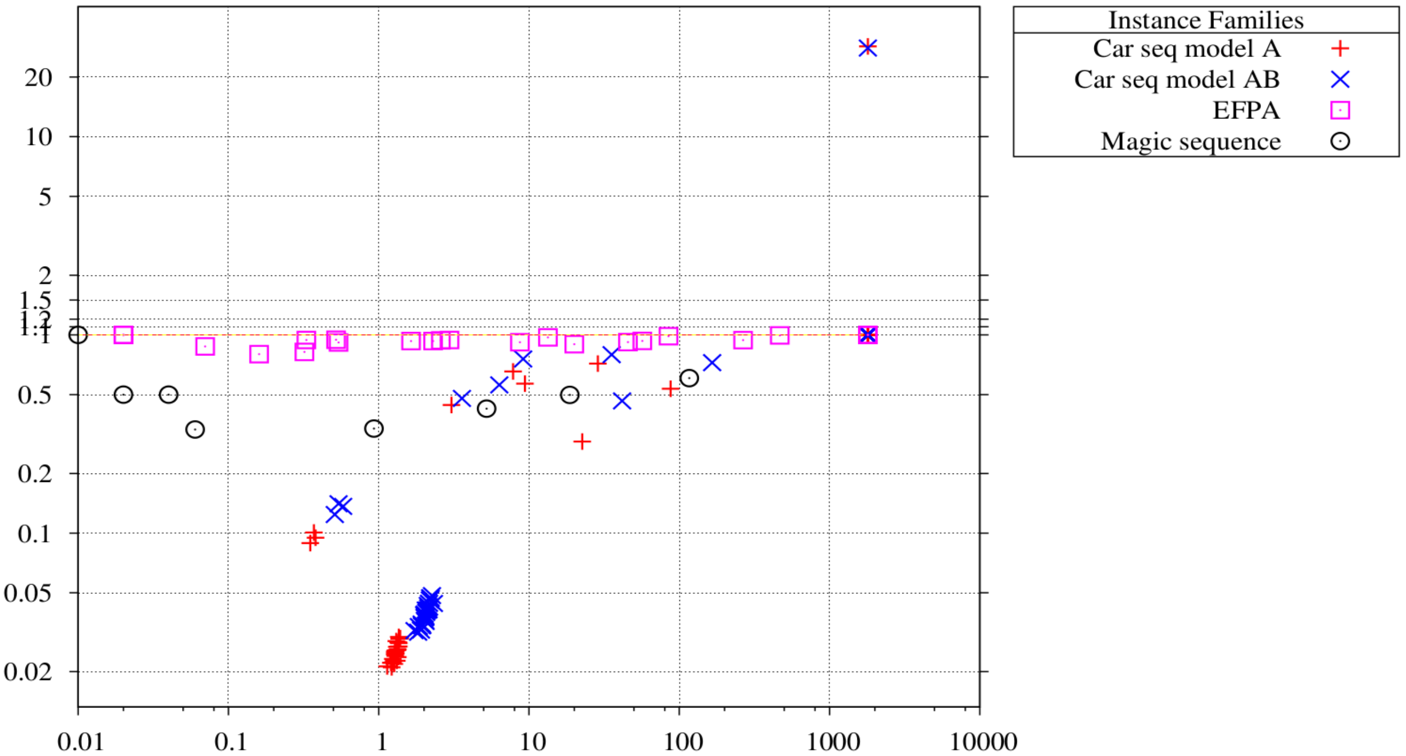
Simple vs Sum

- Sum gives huge improvement for *magic sequence* problem – very unusual structure
- Overall, sum usually worthwhile



Sum vs Flow

- Flow – for each value:
  - Find maximal flows that maximise and minimise occurrences of the value
  - Solves two extra instances within 30 mins



Other Options for (E)GCC

- Bound or Range Consistency propagators
  - Other points on the time/strength tradeoff
  - Not investigated here
- Decomposition (Bessiere et al)
  - nd^2+d^2 extra variables, Range Consistency
  - Exposes the internal state of EGCC
    - Could manually add implied constraints
    - Learning CP solver

Conclusions

- EGCC is one of the key constraints in CP
- Empirical survey of algorithms and optimizations for target variables and cardinalities
  - Some findings go against big-O complexity
- More than 4x improvement from optimizations
  - Same search tree, whole cost of solver
- One new optimisation was very worthwhile
  - Dynamic Partitioning, may apply elsewhere