On the application of graph colouring techniques in round-robin sports scheduling

Theory LANCS Practice

Rhyd Lewis and Jonathan Thompson

School of Mathematics,
Cardiff University,
LANCS Initiative

A case study at the Welsh Rugby Union

Round-robin scheduling: the basics

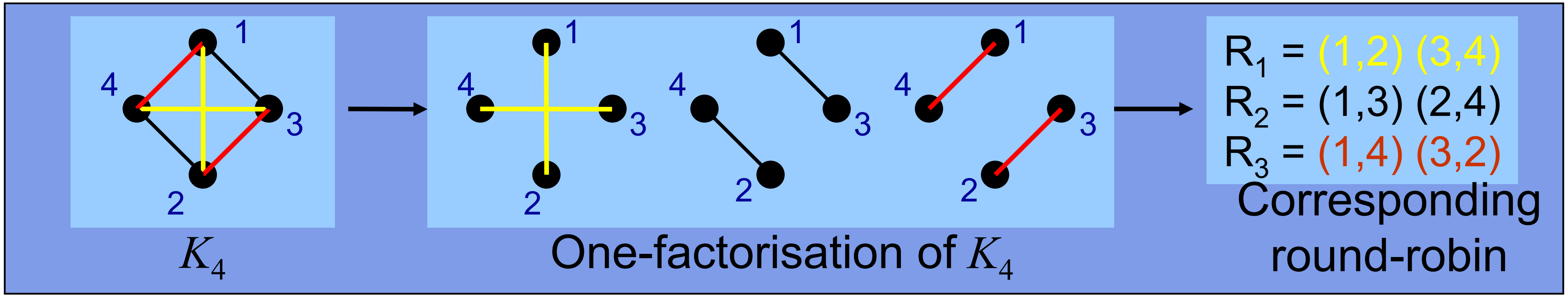
R1 = (0,1), (2,6), (3,5), (4,7)
R2 = (0,2), (1,7), (3,6), (4,5)
R3 = (0,3), (1,2), (4,6), (5,7)
R4 = (0,4), (1,3), (2,7), (5,6)
R5 = (0,5), (1,4), (2,3), (6,7)
R6 = (0,6), (1,5), (2,4), (3,7)
R7 = (0,7), (1,6), (2,5), (3,4)
R8 = (1,0), (6,2), (5,3), (7,4)
R9 = (2,0), (7,1), (6,3), (5,4)
R10 = (3,0), (2,1), (6,4), (7,5)
R11 = (4,0), (3,1), (7,2), (6,5)
R12 = (5,0), (4,1), (3,2), (7,6)
R13 = (6,0), (5,1), (4,2), (7,3)
R14 = (7,0), (6,1), (5,2), (4,3)

Mirrored Schedule for n = 8, m = 2

- We have n teams, and all teams play all others m times in $m(n-1)$ rounds
- Typically, m = 1 or 2
- Linear algorithms exist for producing such schedules such as the greedy **polygon**, and canonical methods
- These "standard solutions" are actually isomorphic
- However, additional constraints will often be imposed that may not fit with such schedules:
  - Home/away assignments
  - Minimisation of "carryover"
  - Broadcaster demands
  - Stadium sharing
  - Mirroring
  - Minimisation of travelling distances.

Number of RR schedules

- The number of distinct round robin schedules is linked to the number of one-factorisations of a complete graph K_n

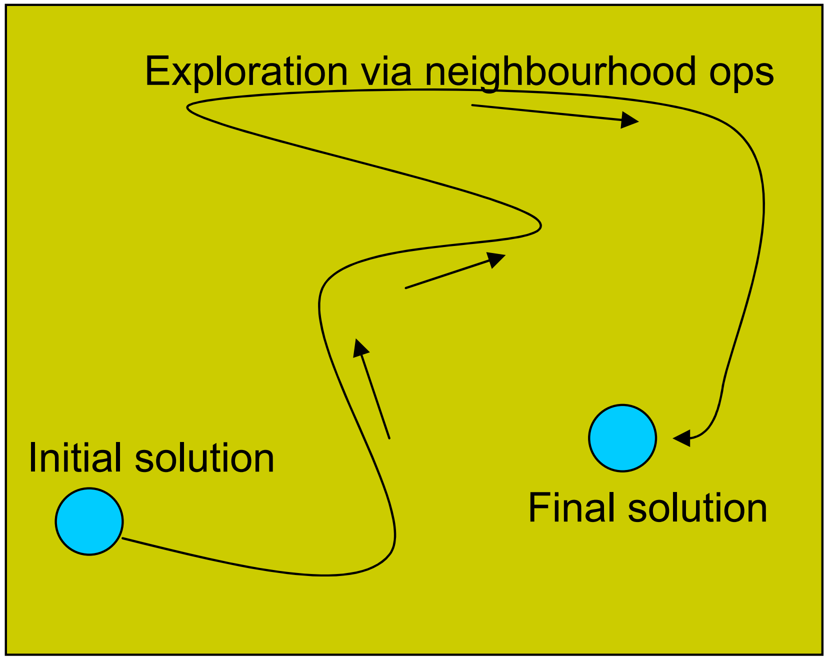


- This figure is thought to increase exponentially with n:
  - n = 2,4,6 : 1 one-factorisation
  - 8 : 6 ``
  - 10 : 396 ``
  - 12 : 526,915,620 (Dinitz et al. 1994)
  - 14+ : ???
- Number of one-factorisations < num round-robins.

Solution methods

- Some formulations are easy to solve in isolation
  - Optimisation of home/away pattern of each team (de Werra, 1988)
  - Maximisation/minimisation of carry-over (for certain *n*'s) (Russell, 1980; Anderson 1999)
- Other formulations (of which there are many) can be more tricky…
  - What about other constraints & combinations of constraints
  - What if these constraints conflict?
  - ILP and CP techniques have been used in the past (though they can sometimes show scaling-up issues)
  - Metaheuristics also show considerable promise …

Metaheuristics for RR Scheduling: Basic strategy

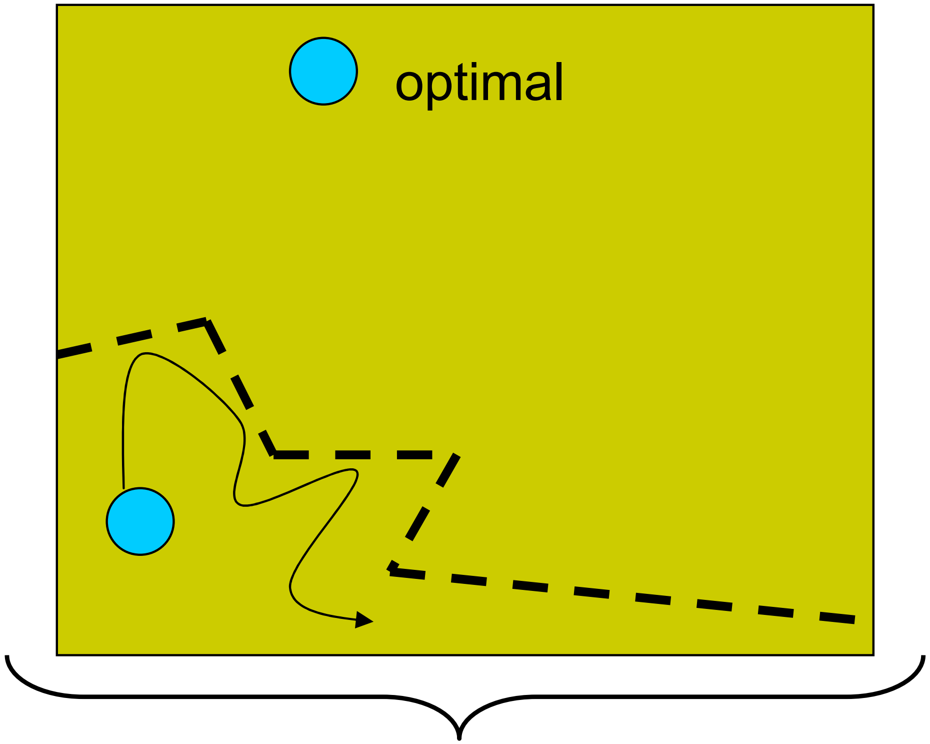


Space of all valid round robins of a particular n

- Define cost function and move operators
- Produce initial solution(s)
- **`while`** (**`not`** stopping criteria)
  - Explore search space attempting to optimise cost
- **`end while`**

- Global optimum not guaranteed (but run-times can be)
- Adaptable to various formulations (just change the cost function)
- Issues:
  - "Standard solutions" often in particular, disconnected regions and show difficulties with existing search operators
  - What if additional "hard constraints" are also considered

Metaheuristics for RR Scheduling: Basic strategy

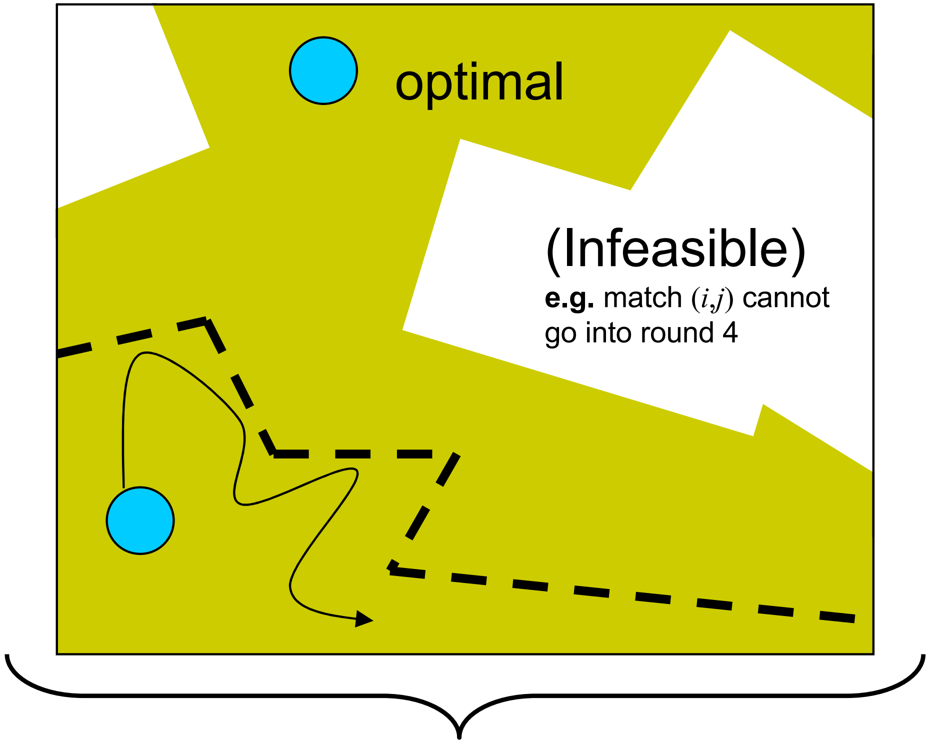


Space of all valid round robins of a particular n

- Define cost function and move operators
- Produce initial solution(s)
- **`while`** (**`not`** stopping criteria)
    - Explore search space attempting to optimise cost
- **`end while`**

- Global optimum not guaranteed (but run-times can be)
- Adaptable to various formulations (just change the cost function)
- Issues:
    - "Standard solutions" often in particular, disconnected regions and show difficulties with existing search operators
    - What if additional "hard constraints" are also considered

Metaheuristics for RR Scheduling: Basic strategy



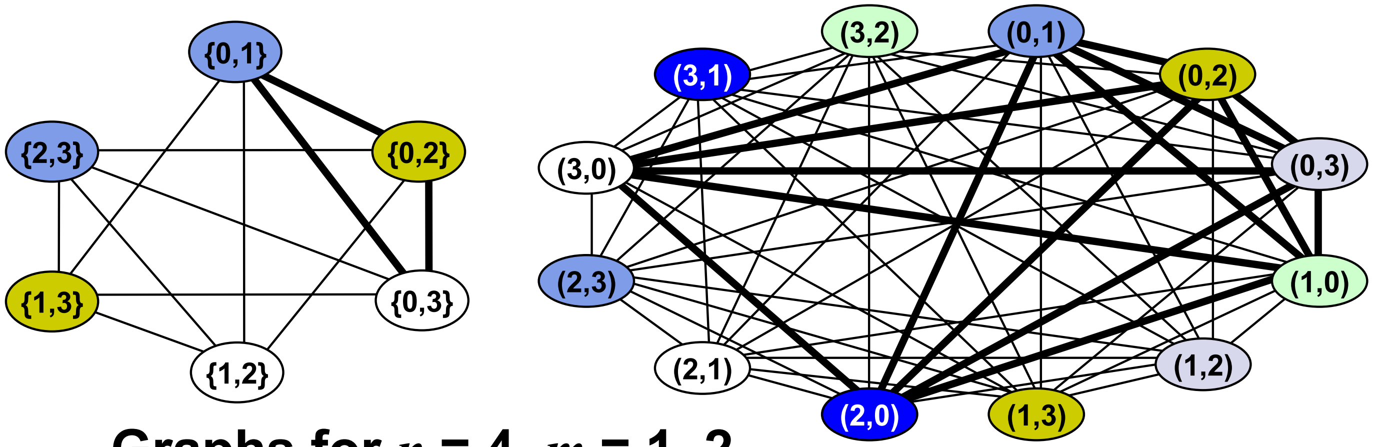
Space of all valid round robins of a particular n

- Define cost function and move operators
- Produce initial solution(s)
- **`while`** (**`not`** stopping criteria)
    - Explore search space attempting to optimise cost
- **`end while`**

- Global optimum not guaranteed (but run-times can be)
- Adaptable to various formulations (just change the cost function)
- Issues:
    - "Standard solutions" often in particular, disconnected regions and show difficulties with existing search operators
    - What if additional "hard constraints" are also considered

RR scheduling via graph colouring

- For a particular n and m:



Graphs for n = 4, m = 1, 2

- One node for each match
- Edges between nodes with a common team

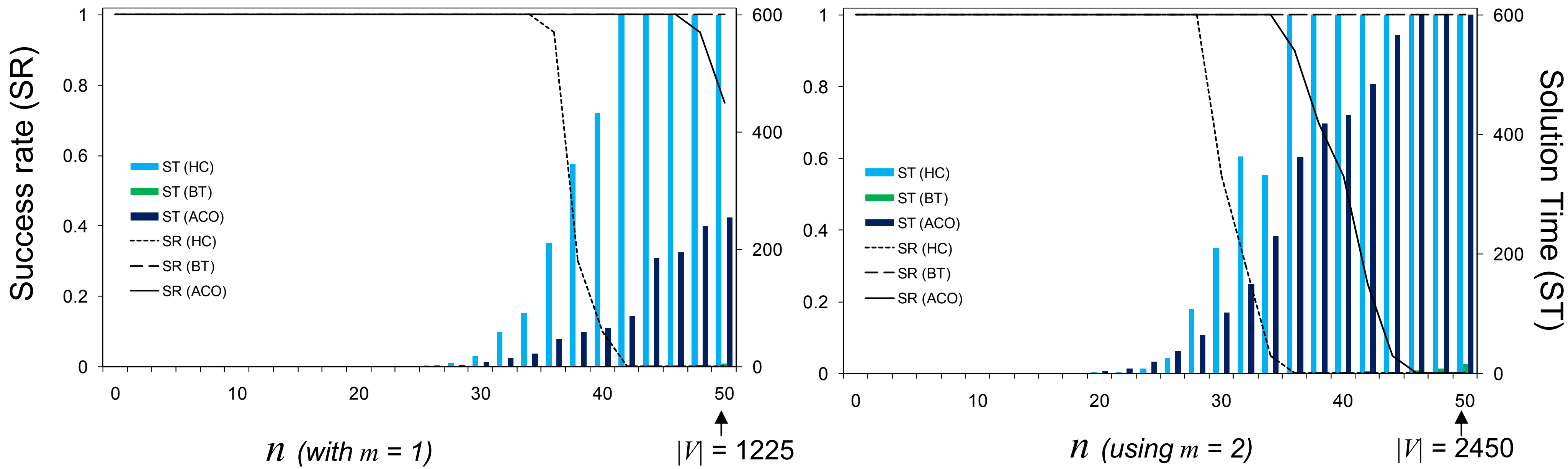
- All nodes equal in degree
- Equipartite
- Interconnected cliques

colour the nodes of the graph using $k = m(n - 1)$ colours such that adjacent nodes have different colours.

- Graph colouring methods might be able to produce RR solutions beyond the "standard" ones
- Such graphs are **flat**: but are they difficult to colour?

Are such graphs “hard” to colour?

- Three different algorithms used to try and solve problems of various size.
  - Heuristic backtracking approach (BT)
  - ACO + Local search approach (ACO)
  - Stochastic Hill Climbing approach (HC)
- Success Rate and Solution Times were recorded
- The backtracking approach seems the most successful.



Are such graphs “hard” to colour?

- We also implemented an IP formulation to run with **MP-Xpress**: Given $G(V,E)$, k, and the binary variables:

$$x_{vj} = 1 \text{ if vertex } v \text{ assigned to colour } j$$
$$= 0 \text{ otherwise}$$

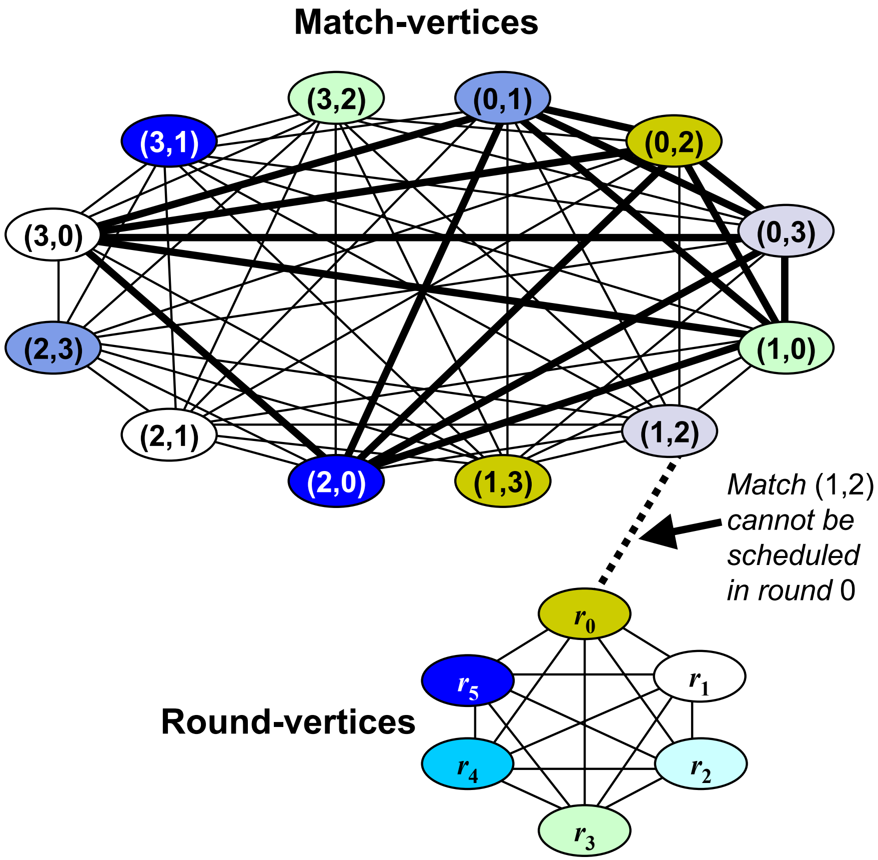
where $1 \leq j \leq k$ and $v \in V$, find an assignment such that:

$$x_{uj} + x_{vj} \leq 1 \quad \forall (u,v) \in E, \quad \forall j$$
$$\sum_{j=1}^{k} x_{vj} = 1 \quad \forall v \in V$$
$$\sum_{v \in V} x_{vj} = \frac{|V|}{k} \quad \forall j$$

- **Less Successful:** No solutions found for $n > 22$ for $m = 1$ and $n > 14$ for $m = 2$

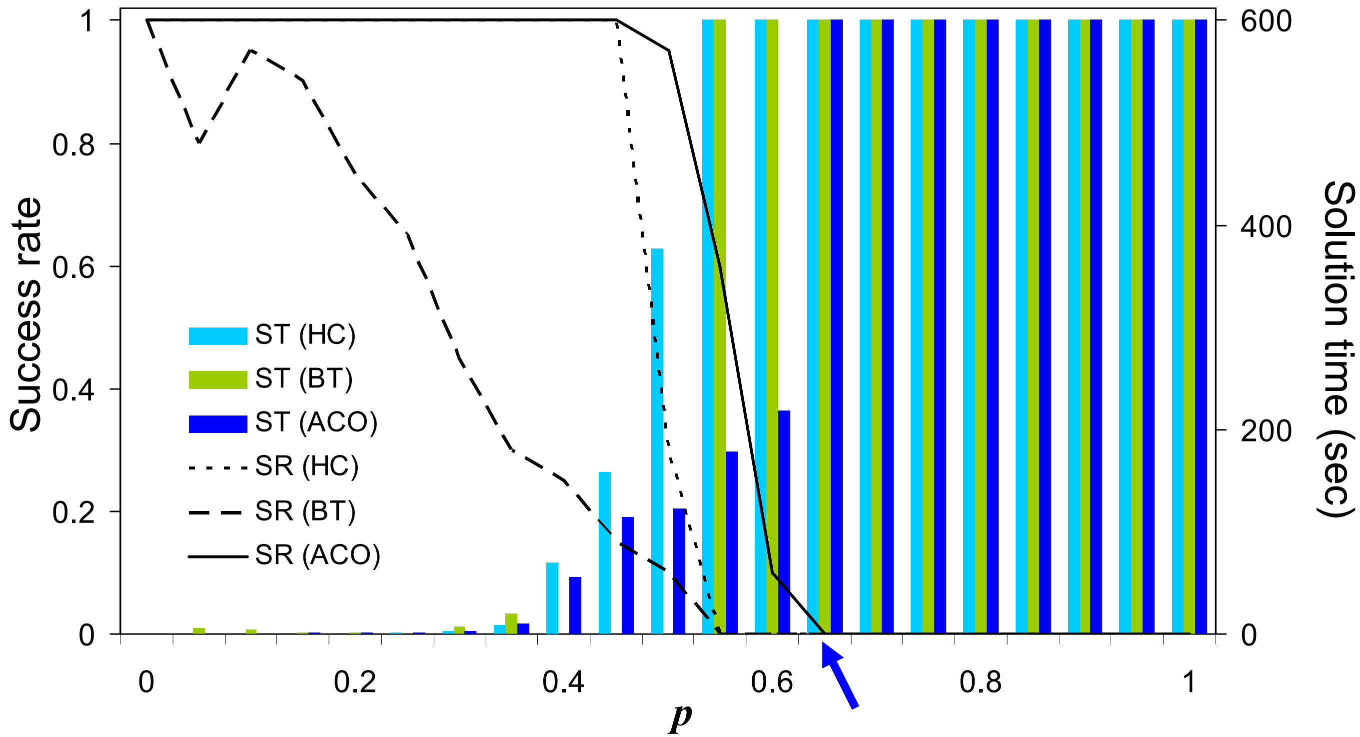
Incorporating additional hard constraints

- A variety of additional hard constraints can be encoded into the graph colouring model. E.g.
  - Match (i, j) cannot occur in round r (unavailability)
  - Match (i, j) should occur in round r (preassignment)
  - Matches (i, j) and (l, m) cannot occur simultaneously
- The model allows us to define move operators that only explore the space of **feasible** RR solutions
- An alternative strategy would be to weight hard constraint violations and hope that they are all satisfied during the search process



Are these new graphs "hard" to colour?

- A problem generator was designed that randomly added constraints of the form "match (i, j) cannot occur in round r".

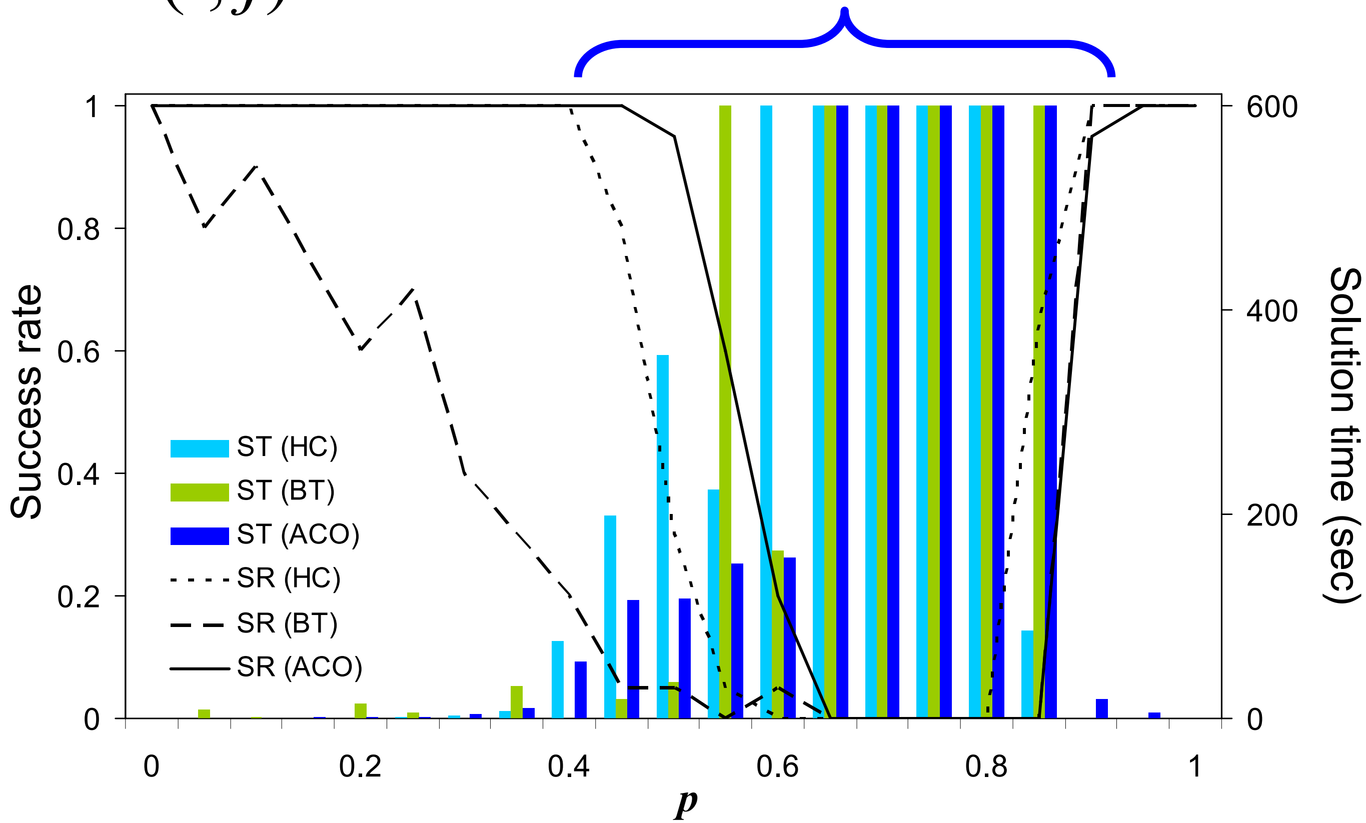


<u>Results with n = 16 , m = 2 ($|V|$ = 240)</u>

- HC and HCO algorithms consistently find solutions for $p < 0.6$ (where matches are only feasible in approx. 40% of rounds)
- Beyond this point solutions may not exist anyway
- Point of "unsolvability" tends to move left with larger problems

Are these new graphs "hard" to colour?

- A problem generator was designed that randomly added constraints of the form "match (i, j) cannot occur in round r".

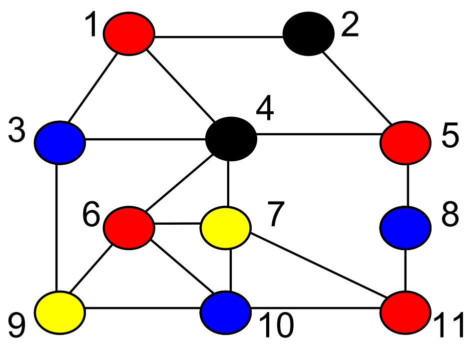


Results with solvable n = 16, m = 2 ($|V|$ = 240)

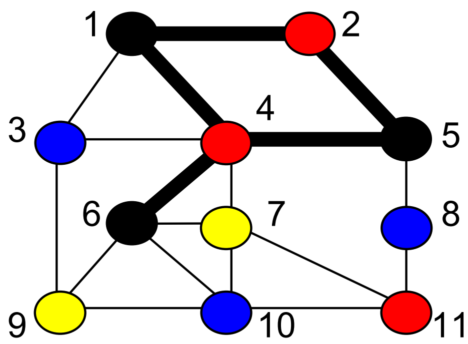
- Similar results with solvable problems, but a phase transition region is clearly visible
- The phase transition tends to widen and deepens with larger problems
- Similar results were also witnessed with other types of constraints such as pre-assignments

Exploring the space of feasible RR schedules

- A suitable candidate is the Kempe chain neighbourhood operator.
- Moves are guaranteed to preserve validity and feasibility

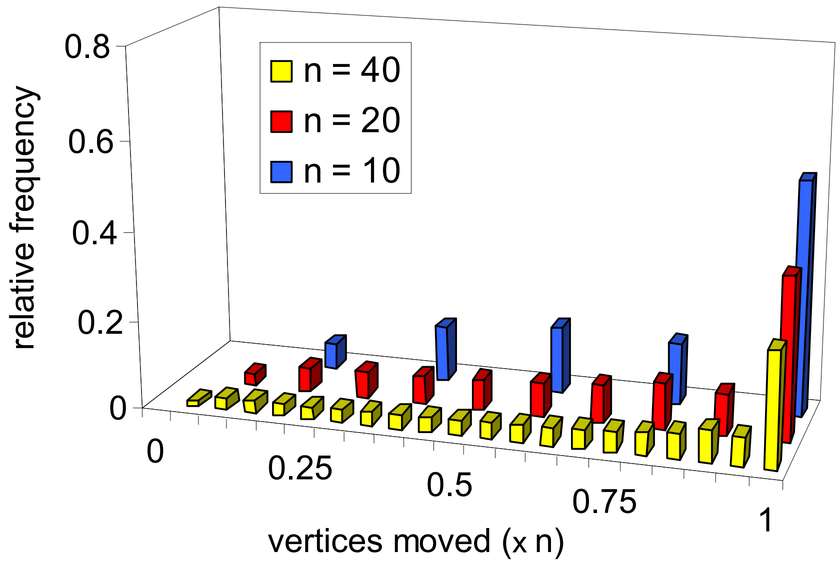


Select a vertex and colour randomly (e.g. vertex 1, black)



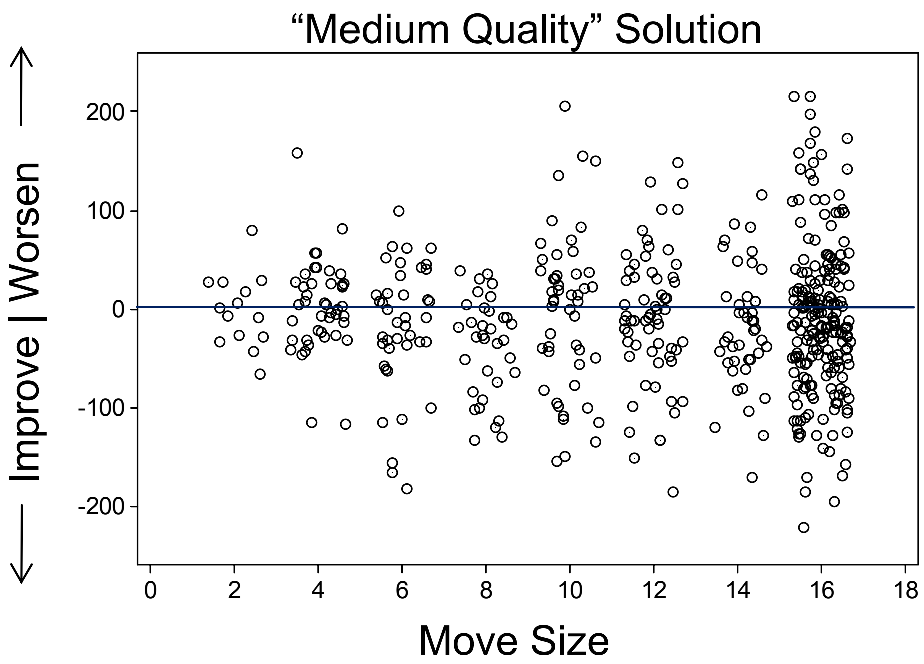
Identify Kempe chain and do colour interchange

- Note that, moves can be of different sizes and seem to arise with particular probabilities
- However, this is not the case if Kempe chains are applied to the “standard solutions” (in these cases all moves are of of size n) *[Perfect 1-factorisations]*

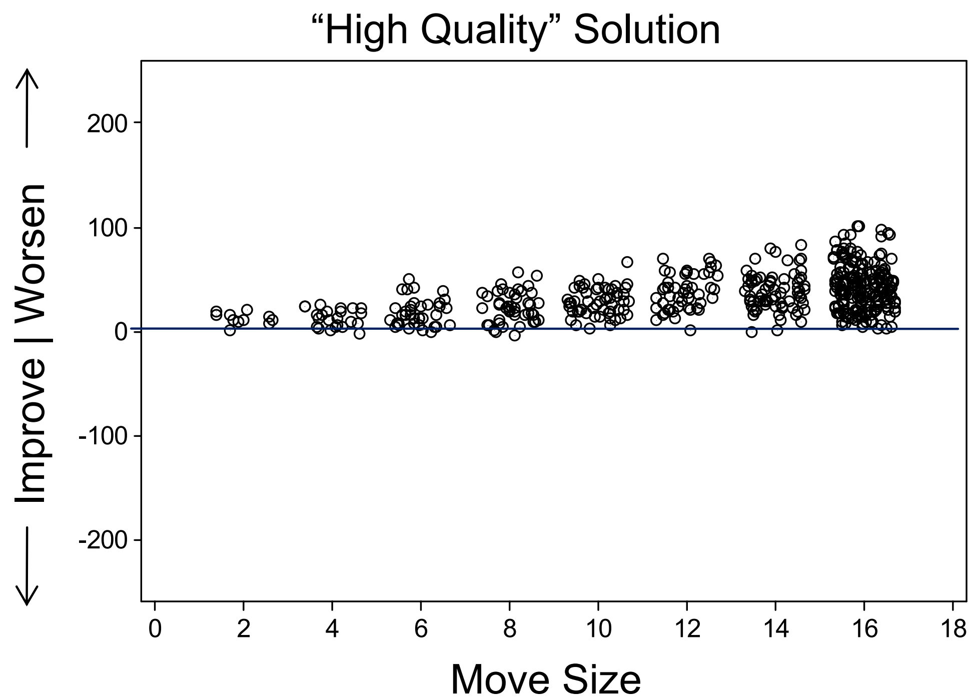


Frequency of move sizes for m = 2

Effect of "Move Size" on Cost



- With a "medium quality" solution, larger moves result in a larger variance in cost
- Approximately half of moves do not worsen a solution



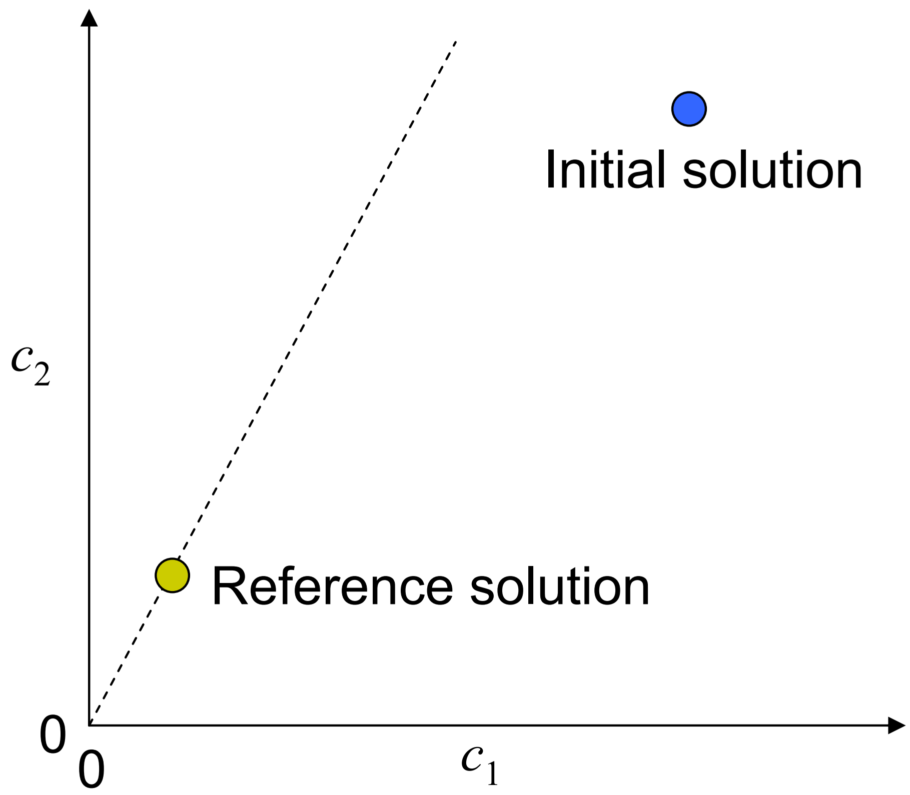
- With a "good" solution, larger moves are associated with larger increases in cost.
- However, larger moves can help to disrupt a solution, helping to escape local optima

Case Study: The Welsh Principality Premiership

- Top domestic rugby union league in Wales: 14 teams to play in a double round-robin tournament
  - Additional Hard Constraints
    - One pair of teams share a stadium
    - Three other teams share stadia with teams from different leagues/sports
    - Local derbies take place on Xmas/Easter weekends
  - Soft constraints
    - SC_1: Teams to play H-A-H-A… as much as possible
    - SC_2: Teams cannot play each other twice within 5 rounds, and should meet in different "halves" of the season.
- It was found that hard constraints could be solved quite easily using the three graph colouring methods (though not with IP formulation).

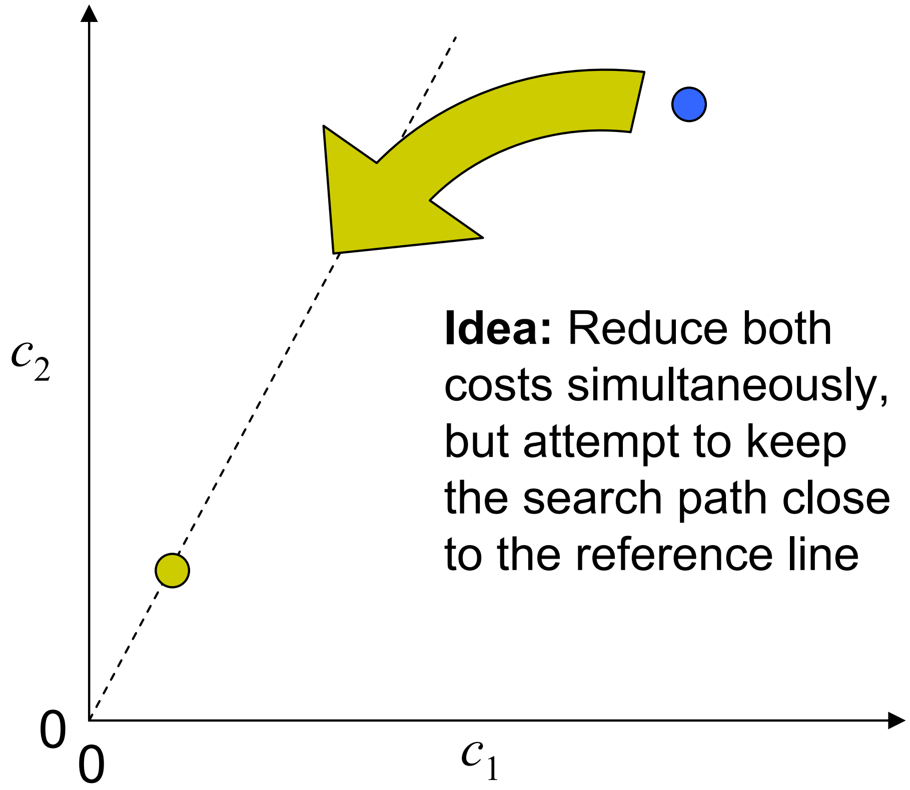
A Multiobjective Approach

- Cost functions c_1 and c_2 used for reflecting level of compliance with SC_1 and SC_2 respectively.
- c_1 and c_2 measure different things: thus a static aggregate cost function not satisfactory.
- Two methods were applied to get around this. The best of these is based on a multi-objective approach of Petrovic and Bykov (2003)



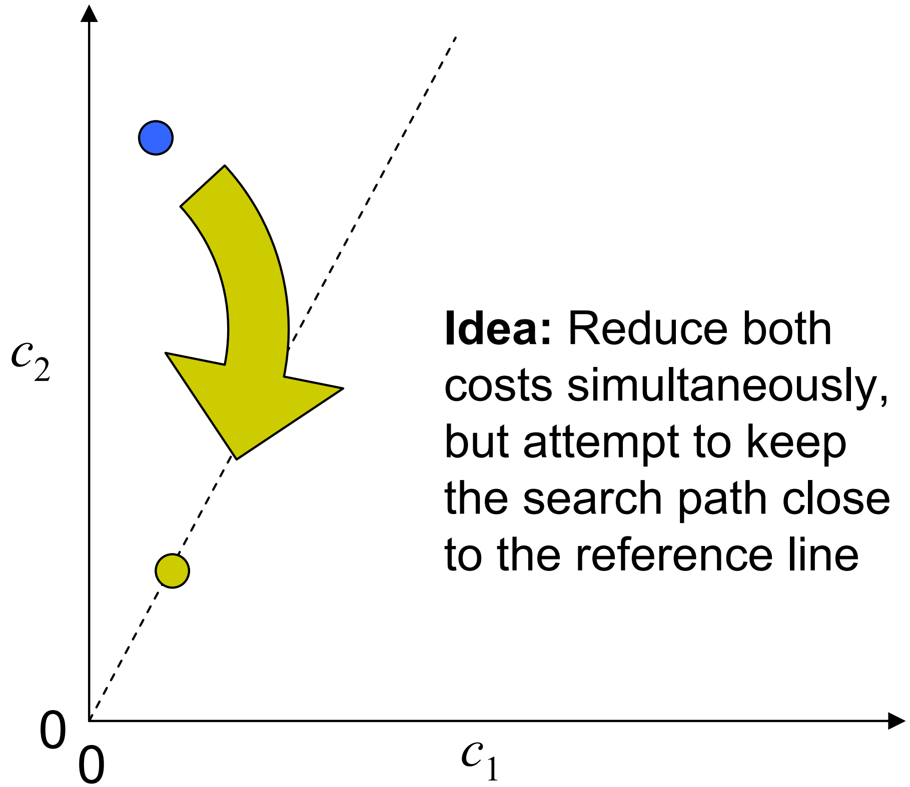
A Multiobjective Approach

- Cost functions c_1 and c_2 used for reflecting level of compliance with SC_1 and SC_2 respectively.
- c_1 and c_2 measure different things: thus a static aggregate cost function not satisfactory.
- Two methods were applied to get around this. The best of these is based on a multi-objective approach of Petrovic and Bykov (2003)



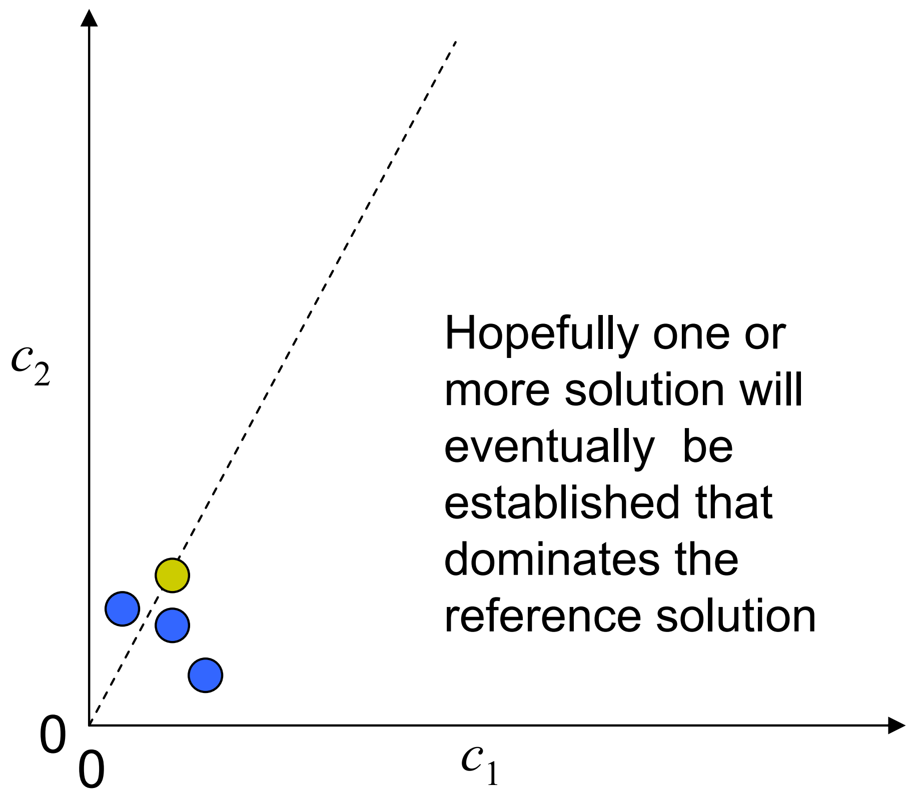
A Multiobjective Approach

- Cost functions c_1 and c_2 used for reflecting level of compliance with SC_1 and SC_2 respectively.
- c_1 and c_2 measure different things: thus a static aggregate cost function not satisfactory.
- Two methods were applied to get around this. The best of these is based on a multi-objective approach of Petrovic and Bykov (2003)



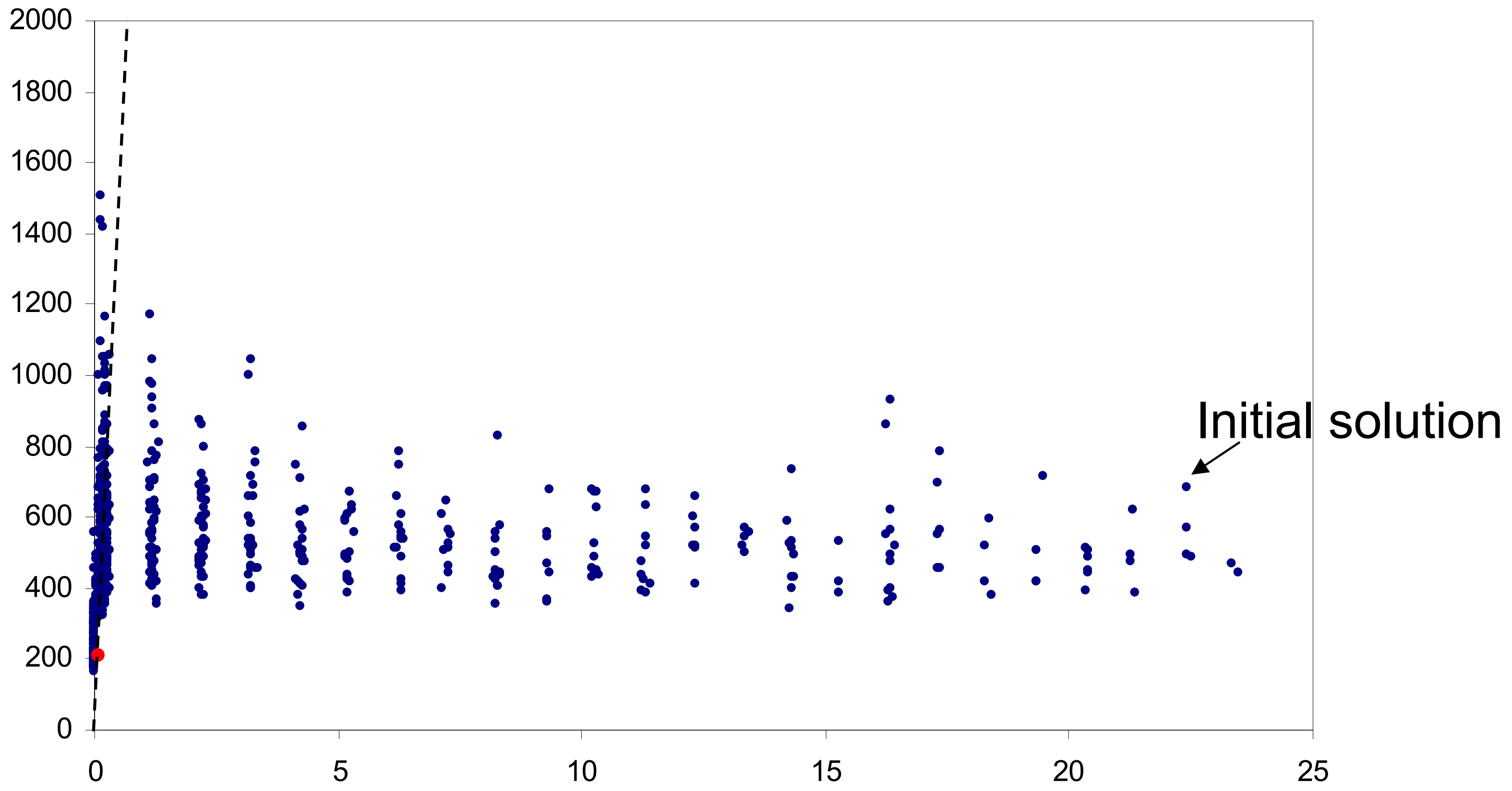
A Multiobjective Approach

- Cost functions c_1 and c_2 used for reflecting level of compliance with SC_1 and SC_2 respectively.
- c_1 and c_2 measure different things: thus a static aggregate cost function not satisfactory.
- Two methods were applied to get around this. The best of these is based on a multi-objective approach of Petrovic and Bykov (2003)



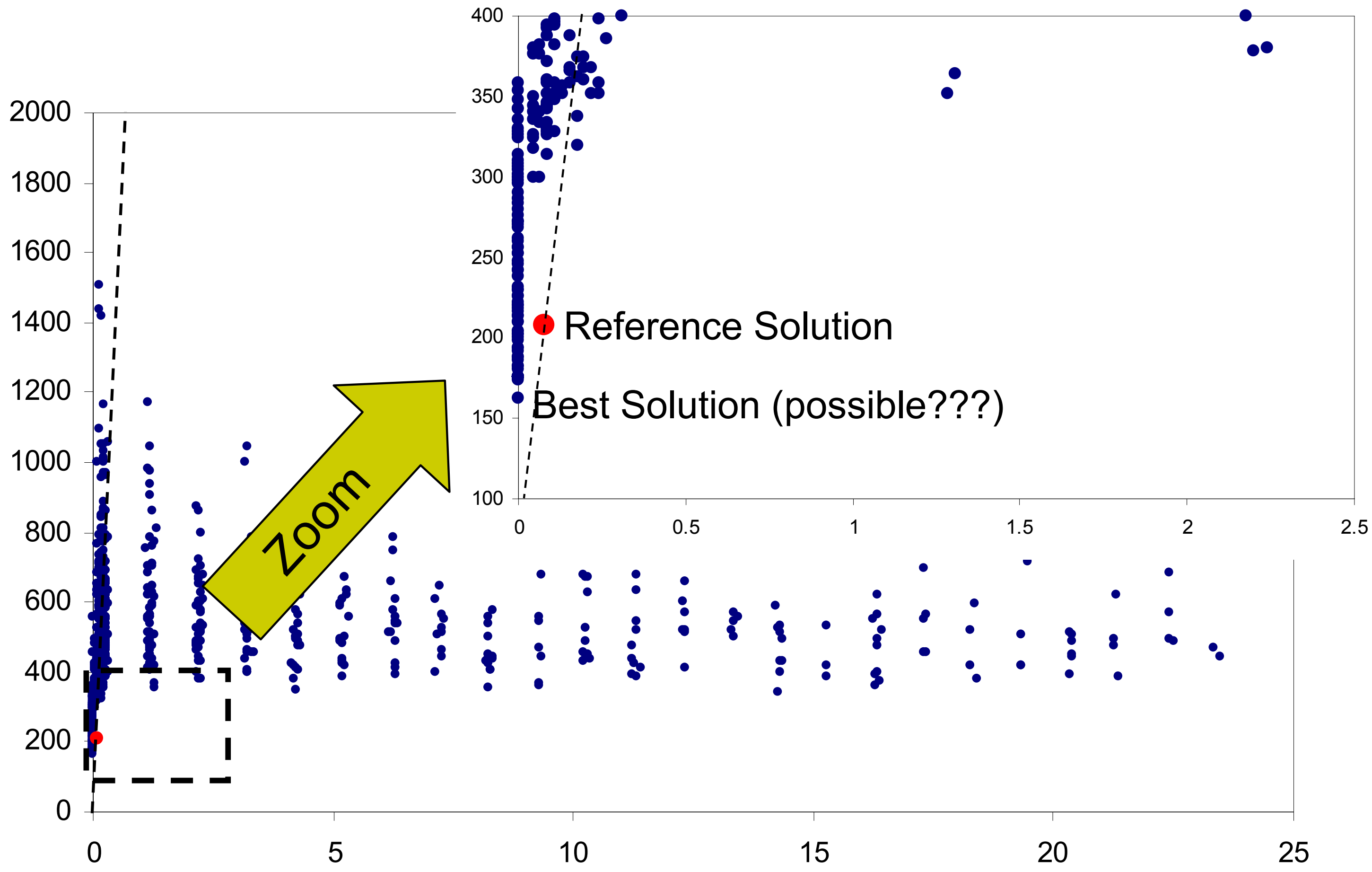
Example Run with real-world data

- Feasible solution found quickly (<10 CPU sec). Approx 10 min granted to soft constraint optimiser



Example Run with real-world data

- Feasible solution found quickly (<10 CPU sec). Approx 10 min granted to soft constraint optimiser



Conclusions

- Graph colouring methods have the potential to produce feasible round-robin schedules, often in the presence of a large number of additional constraints
- Generic search operators can also be defined with this model allowing the exploration of the space of **feasible** round robins (a subset of all **valid** RRs)
- Over 98% of solutions produced using our methods dominated those manually produced by the league administrators.
- Methods were also seen to perform well on larger/smaller instances.

- **Ongoing issues:**
  - Are other graph colouring methods more suitable for such applications?
  - Deeper understanding of search space connectivity is needed.
  - Are other neighbourhood operators applicable?
  - Where else can these solution methods be applied?