Fitness modelling
for better optimisation and decision making

Sandy Brownlee
www.cs.stir.ac.uk/~sbr
sbr@cs.stir.ac.uk

Who am I?

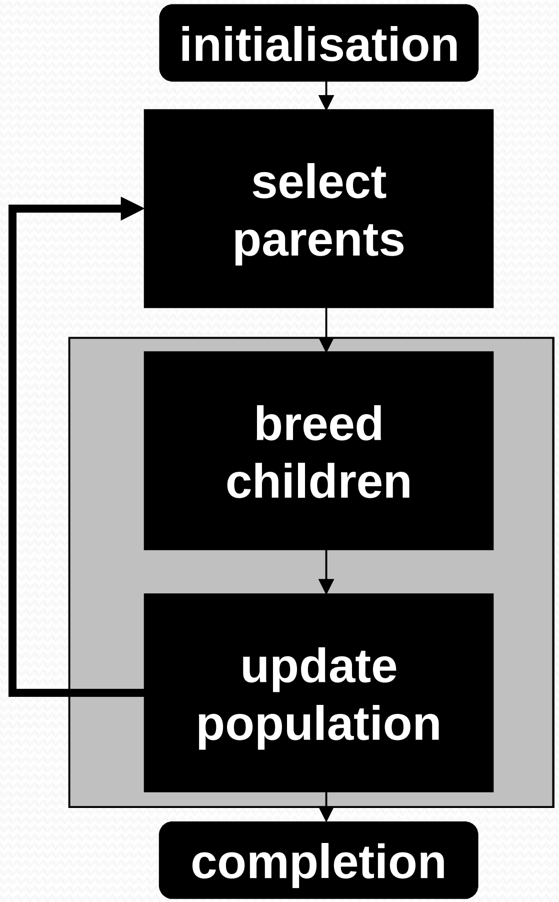
- Started in July, member of CHORDS group
- Airport operations optimisation
- Previous:
  - RA Loughborough University: multi-objective building design optimisation
  - PhD Robert Gordon University: fitness modelling, EDAs

Outline

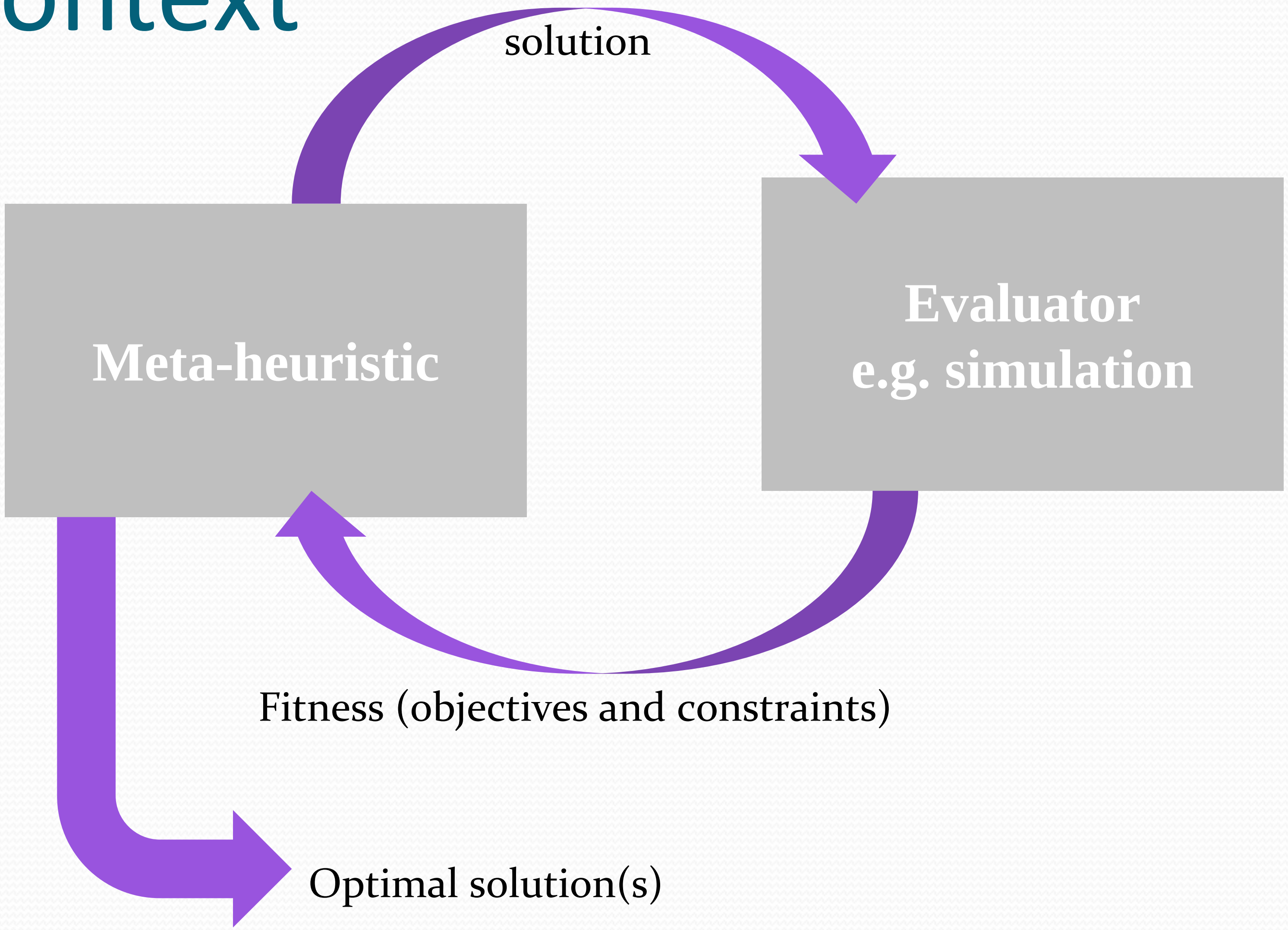
- Context: optimisation, meta-heuristics, evolutionary algorithms
- Fitness models, the MFM, and DEUM EDA
- Speedup – FM as surrogates
- Decision support – mining FM
- What makes a good model? – and the broader impact

Context

- Meta-heuristics (e.g. EA)
- Explores the space of solutions to a problem (typically quite big)
- Evolution is random, but guided by *fitness* (objectives and constraints)
- Solutions can look quite different: set of bits, integers, real values, trees, programs…



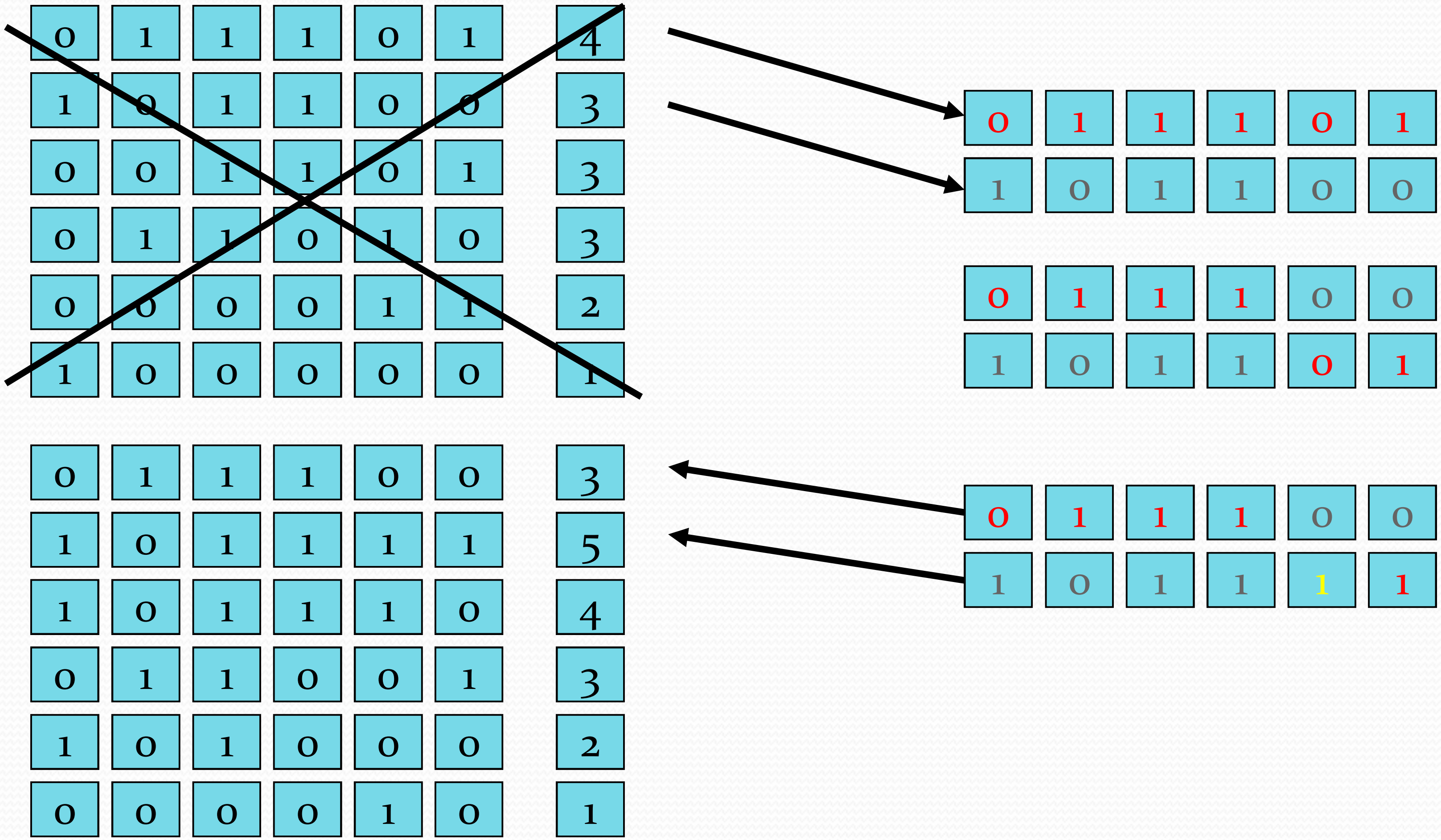
Context


solution
Meta-heuristic
Evaluator
e.g. simulation
Fitness (objectives and constraints)
Optimal solution(s)

Fitness, objectives, constraints

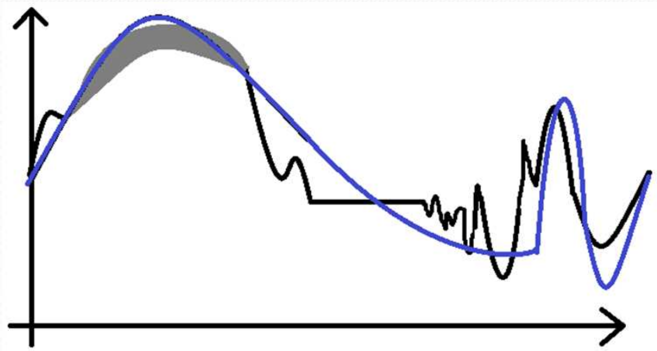
- "Fitness" / "fitness function": how algorithm compares solutions
- Objectives: things to minimise / maximise
- Constraints: pass / fail particular solutions
- A fitness model attempts to approximate all or some of the above

Single Objective EA Example



Fitness models

- Try to estimate some or all of the fitness landscape
- a.k.a meta-models, fitness approximations, surrogates , “model of the model”
- Several uses:
  - Reduce cost associated with evaluations
  - Overcome difficult search landscape
    - Noise, multi-modality, plateaux
  - Used if no explicit fitness function (e.g. evolutionary art, real-world measurement)
- Many approaches:
  - Neural networks, support vector machines, Kriging, database, probabilistic model

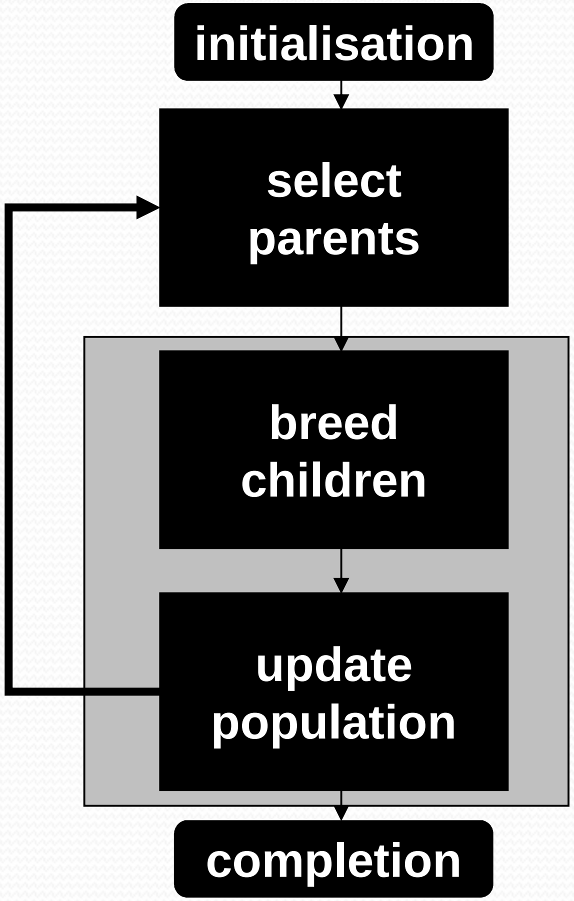


Example fitness models

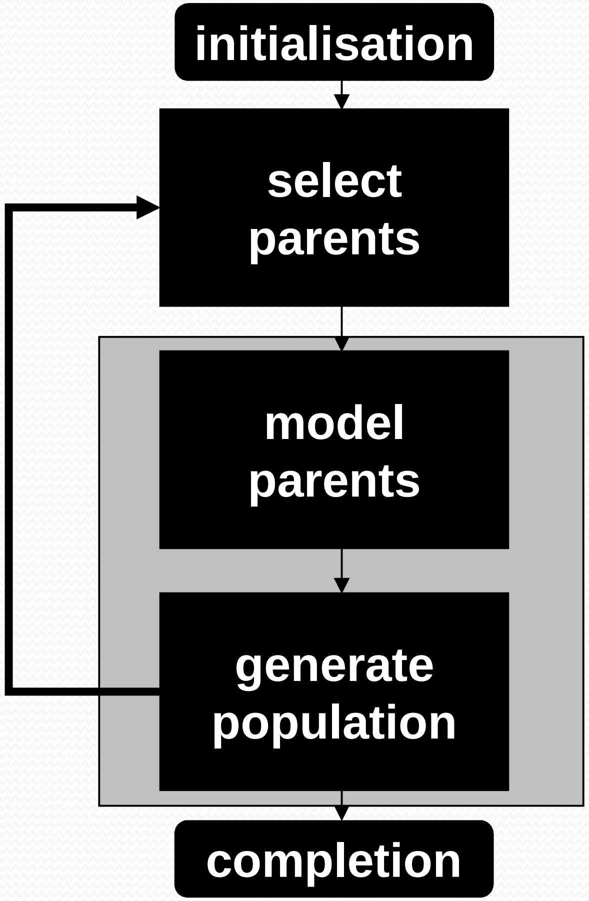
- “A couple” now rounded to “approximately one”
- Markov network fitness model (MFM)
  - Targeted at binary / bit-string representations
- Radial basis function network (RBFN)
  - Targeted at mixed representation (continuous & discrete variables)

Model 1 : MFM

- Originally developed as part of DEUM EDA



GA



EDA

Probabilistic models

- Solution x - a collection of random variables
- Model distribution of x as a joint probability distribution (j.p.d.)
- Could simply use marginal probabilities of variables
- What if there are dependencies between variables?

x_1	x_2	x_3	x_4	x_5	x_6

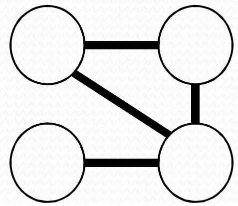
$$x = x_1, x_2, \ldots, x_n$$

$$p(x) = p(x_1, x_2, \ldots, x_n)$$

0	1	1	1	0	0
1	0	1	1	0	0
0	0	1	1	0	1
0	1	1	0	1	0

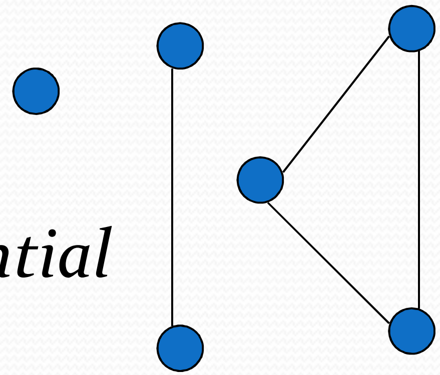
0.25	0.5	1	0.75	0.25	0.25

Markov Network



- An undirected probabilistic graphical model
  - Contrast with Bayesian network (directed graph)
  - Representation of the joint probability distribution
  - Variables become nodes on a graph
  - Edges represent dependencies between variables
- Markovianity property
  - distribution of a variable determined by its neighbours
- Hammersley-Clifford theorem
  - j.p.d. factorises as a **Gibbs distribution**, defined over the cliques of the graph
  - (a clique is a set of mutually neighbouring variables)

Markov Network



- Cliques have energy, defined in a *clique potential function*
- MN describes energy U(x) as sum of clique potentials
- In DEUM, Gibbs distribution of MN is equated to mass distribution of fitness in population

$$p(x) = \frac{f(x)}{\sum_y f(y)} \equiv \frac{e^{-U(x)/T}}{\sum_y e^{-U(y)/T}}$$

$$-\ln(f(x)) = U(x)/T$$

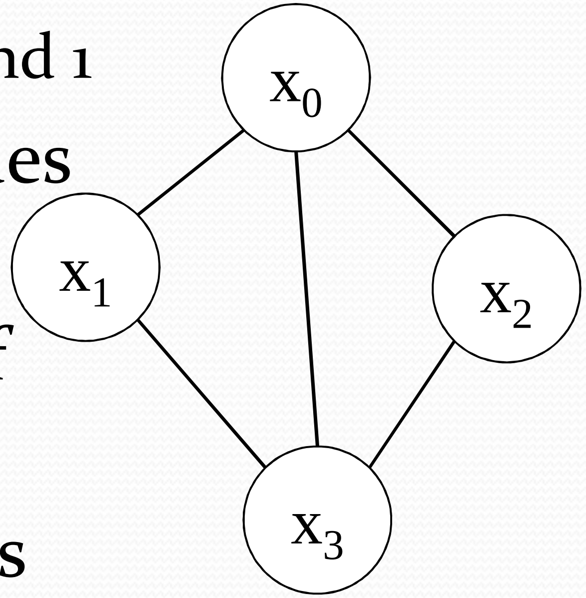
- Energy has negative log relationship to probability, so minimise U to maximise f
- (CPFs correspond to Walsh functions)

Markov network example

- Model can be represented by:

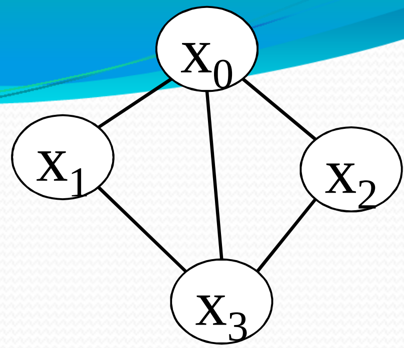
$$\alpha_0 x_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \alpha_{01} x_0 x_1 + \alpha_{02} x_0 x_2 + \alpha_{03} x_0 x_3 \\ \alpha_{13} x_1 x_3 + \alpha_{23} x_2 x_3 + \alpha_{013} x_0 x_1 x_3 + \alpha_{023} x_0 x_2 x_3 + c = -\ln(f(x))$$

  - Variables are -1 and +1 instead of 0 and 1



- Build a set of equations using values from population
- Use least squares fit to solve set of equations and estimate α values
- Also need to determine the cliques (structure)

Markov network example



Sample points / MN parameters / optimal solution

1011 f=1

$$(1)\alpha_0 + (-1)\alpha_1 + (1)\alpha_2 + (1)\alpha_3 + (1)(-1)\alpha_{01} + (1)(1)\alpha_{02} + (1)(1)\alpha_{03} + (-1)(1)\alpha_{13} + (1)(1)\alpha_{23} + (1)(-1)(1)\alpha_{013} + (1)(1)(1)\alpha_{023} + c = -\ln(1)$$

1111 f=4

$$(1)\alpha_0 + (1)\alpha_1 + (1)\alpha_2 + (1)\alpha_3 + (1)(1)\alpha_{01} + (1)(1)\alpha_{02} + (1)(1)\alpha_{03} + (1)(1)\alpha_{13} + (1)(1)\alpha_{23} + (1)(1)(1)\alpha_{013} + (1)(1)(1)\alpha_{023} + c = -\ln(4)$$

1001 f=1

$$(1)\alpha_0 + (-1)\alpha_1 + (-1)\alpha_2 + (1)\alpha_3 + (1)(-1)\alpha_{01} + (1)(-1)\alpha_{02} + (1)(1)\alpha_{03} + (-1)(1)\alpha_{13} + (-1)(1)\alpha_{23} + (1)(-1)(1)\alpha_{013} + (1)(-1)(1)\alpha_{023} + c = -\ln(1)$$

1000 f=3

$$(1)\alpha_0 + (-1)\alpha_1 + (-1)\alpha_2 + (-1)\alpha_3 + (1)(-1)\alpha_{01} + (1)(-1)\alpha_{02} + (1)(-1)\alpha_{03} + (-1)(-1)\alpha_{13} + (-1)(-1)\alpha_{23} + (1)(-1)(-1)\alpha_{013} + (1)(-1)(-1)\alpha_{023} + c = -\ln(3)$$

0011 f=2

$$(-1)\alpha_0 + (-1)\alpha_1 + (1)\alpha_2 + (1)\alpha_3 + (-1)(-1)\alpha_{01} + (-1)(1)\alpha_{02} + (-1)(1)\alpha_{03} + (-1)(1)\alpha_{13} + (1)(1)\alpha_{23} + (-1)(-1)(1)\alpha_{013} + (-1)(1)(1)\alpha_{023} + c = -\ln(2)$$

α_0=-0.38	α_1=0.16	α_2=0.02	α_3=-0.34
α_{01}=-0.07	α_{02}=0.25	α_{03}=-0.11	α_{13}=-0.11
α_{23}=-0.25	α_{013}=-0.34	α_{023}=-0.02	c=-0.61

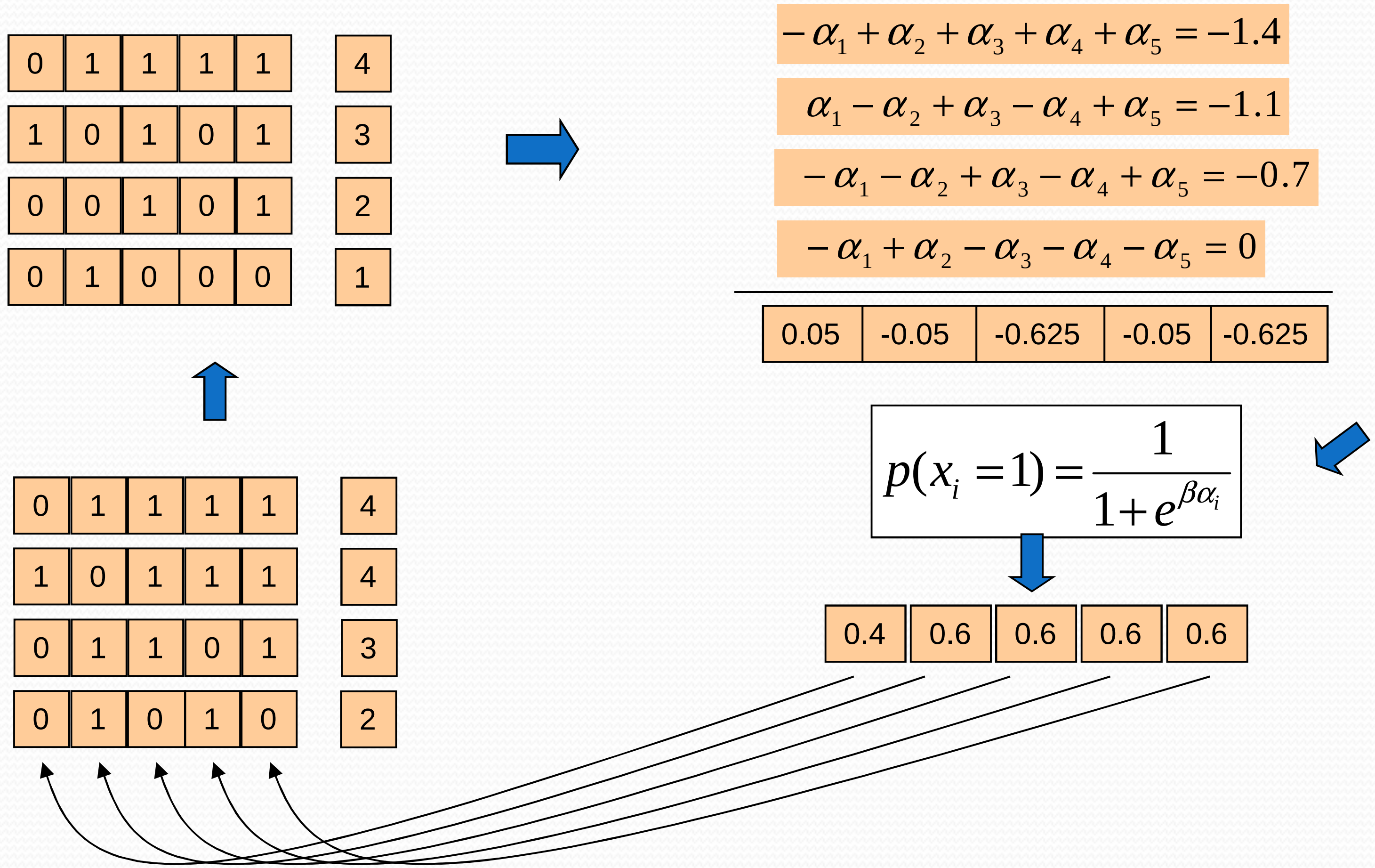
Sampling in DEUM EDA

$$p(x_i = 1) = \frac{1}{1 + e^{2\omega_i / T}} \qquad p(x_i = -1) = \frac{1}{1 + e^{-2\omega_i / T}}$$

- Decreasing temperature *T* cools probability to either 1 or 0 depending upon sign and value of ω
- Sampling the probability gives a value for a particular variable - this forms the basis for the optimisation algorithm $\mathrm{DEUM_d}$

Example run of DEUM$_d$

- Here showing a univariate model



MN Model Predicts Fitness

- Example; for solution X={1011}
- Substitute variable values into energy function and solve:

$$U(x) = \alpha_0 - \alpha_1 + \alpha_2 + \alpha_3 - \alpha_{01} + \alpha_{02} + \alpha_{03} - \alpha_{13} + \alpha_{23} - \alpha_{013} + \alpha_{023} + c$$

$$f(x) = e^{-U(x)}$$

- Hence, the Markov network Fitness Model

Outline

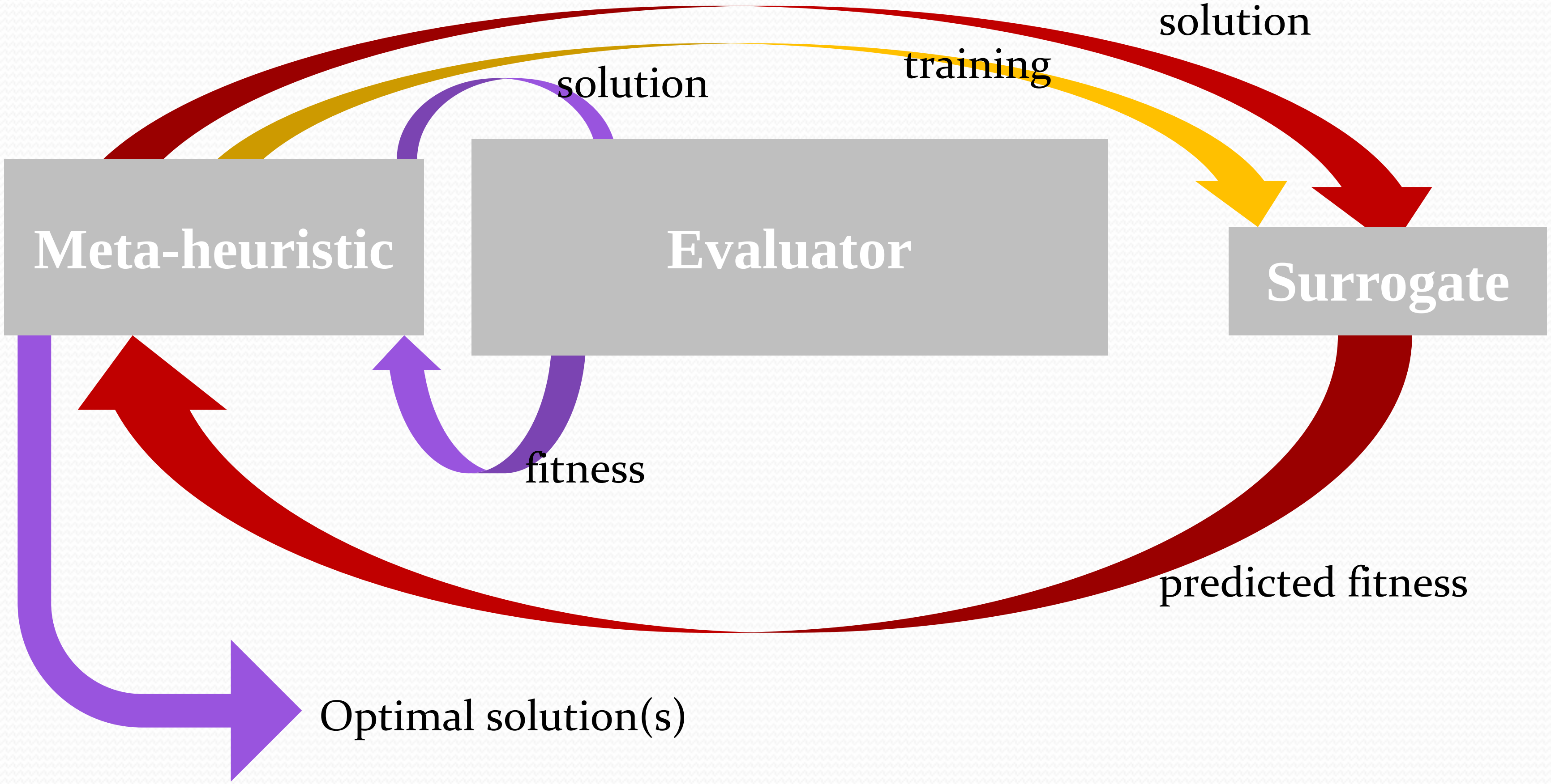
- Context: optimisation, meta-heuristics, evolutionary algorithms
- Fitness models, the MFM, and DEUM EDA
- Speedup – FM as surrogates
- Decision support – mining FM
- What makes a good model? – and the broader impact

FMs as surrogates

- Common use of fitness model is to reduce calls to true fitness function
- Function may be costly: e.g. long run-time or human evaluation
- Two broad approaches:
  - Surrogate FM is trained prior to run and used in place of fitness function
  - "Evolution control": some evaluations are replaced with calls to the surrogate, and surrogate may be updated as run proceeds

Fitness Model as a Surrogate

- Train a model of the fitness function
- Use the model in place of the FF (sometimes)



Example - feature selection

- Feature selection (CBR) - costly fitness function
  - Choose features that best distinguish cases
  - Must run through entire case-base counting whether cases were correctly classified
- GA previously applied to FS
  - Bitstring encoding, 1=selected, 0=not selected
- Uses two public domain datasets:
  - Sonar – 60 features, 208 cases
  - Vehicle – 18 features, 946 cases

MFM-GA

Plain EA

1. Generate random population
2. Compute true *fitness* for members of the population
3. Choose the best ones and recombine them to produce *offspring*
4. Mutate the offspring
5. Repeat 1-5 until we're done

MFM-GA

EA with surrogate

1. Generate random population
2. Every n^{th} generation:
   1. Compute true *fitness* for members of the population
   2. estimate model parameters
3. Otherwise:
   1. Use model to estimate fitness of population
4. Choose the best ones and recombine them to produce *offspring*
5. Mutate the offspring
6. Repeat 1-5 until we're done

Results

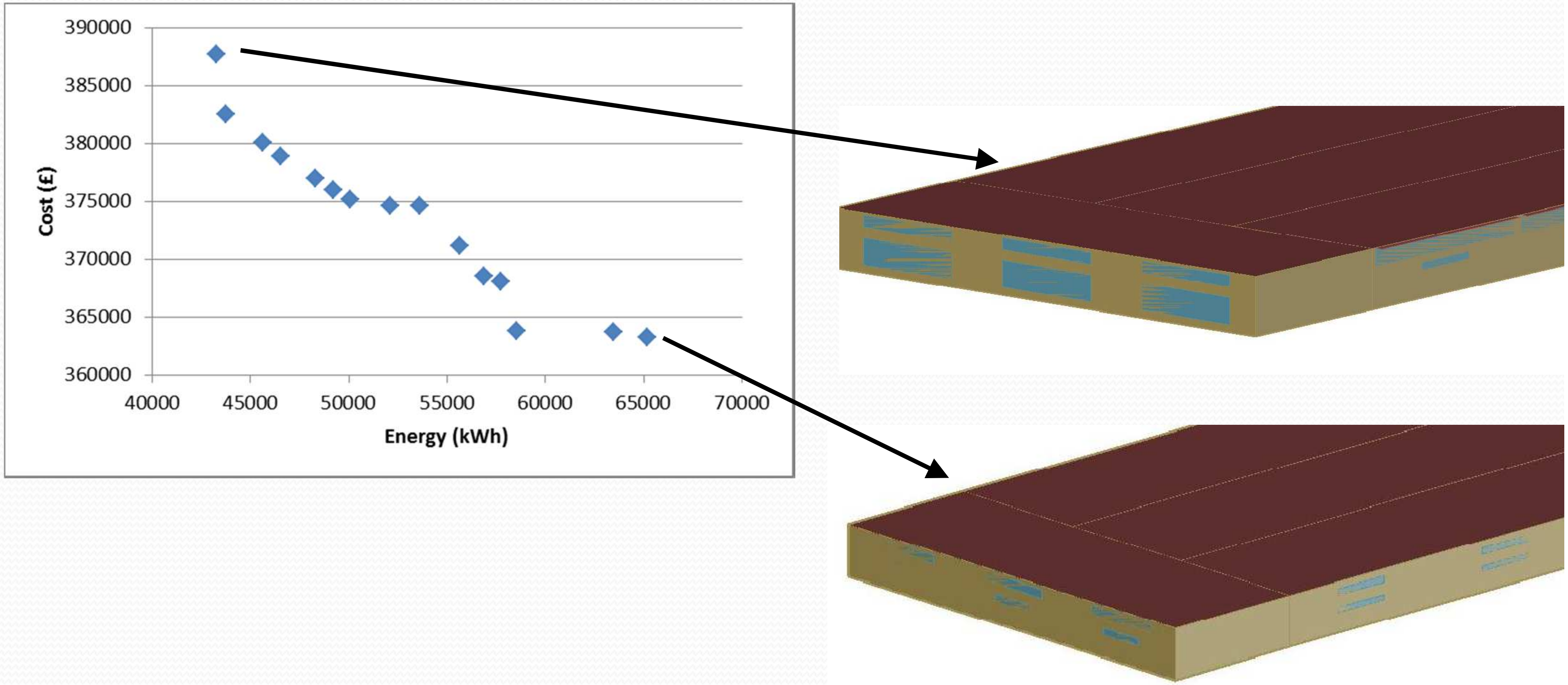
	Sonar		Vehicle	
Algorithm	Best Fitness (SD)	Time (SD)	Best Fitness (SD)	Time (SD)
GA	0.952 (0.010)	796 s (18)	0.756 (0.006)	7778 s (593)
MFM-GA_0	0.910 (0.012) [-0.042 on GA]	147s (11) [0.18 x GA]	0.721 (0.006) [-0.035 on GA]	283 s (43) [0.04 x GA]
MFM-GA_{10}	0.908 (0.015) [-0.044 on GA]	272 s (12) [0.34 x GA]	0.726 (0.010) [-0.030 on GA]	1270 s (78) [0.16 x GA]

Results

- Faster, but reduction in final solution quality
- (still higher fitness than CBR-specific filter selection techniques: information gain, SVM, feature subset evaluation)
- Improved by updating model, with a trade-off in speedup

Building optimisation

- Used RBFN as surrogate, with mixed variable types, multiple objectives and constraints



Building optimisation

Plain EA

1. Generate random population
2. Assign a *fitness* to members of the population
3. Choose the best ones and recombine them to produce *offspring*
4. Mutate the offspring
5. Repeat 1-4 until we're done

Building optimisation

EA with surrogate

1. Generate random population
2. Assign a *fitness* to members of the population
3. Train surrogate
4. Choose the best ones and recombine them to produce too many *offspring*
5. Mutate the offspring
6. Use surrogate to filter out promising offspring
7. Repeat 1-6 until we're done

Results

- Speedup / found higher hypervolume
- NB - constraints need special treatment

Algorithm variant	Hypervolume	p-value	SR (%)	Evals
NSGA-II	0.849 (0.028)	n/a	50	4017
NSGA-II$_c$	0.845 (0.022)	0.969	40	4026
NSGA-II-S	0.856 (0.028)	0.783	83	3817
NSGA-II-S$_c$	0.860 (0.024)	0.338	63	4002
NSGA-II-S$_d$	0.881 (0.031)	< 0.001	83	3184
NSGA-II-S$_{cd}$	0.867 (0.027)	0.034	73	3340

Outline

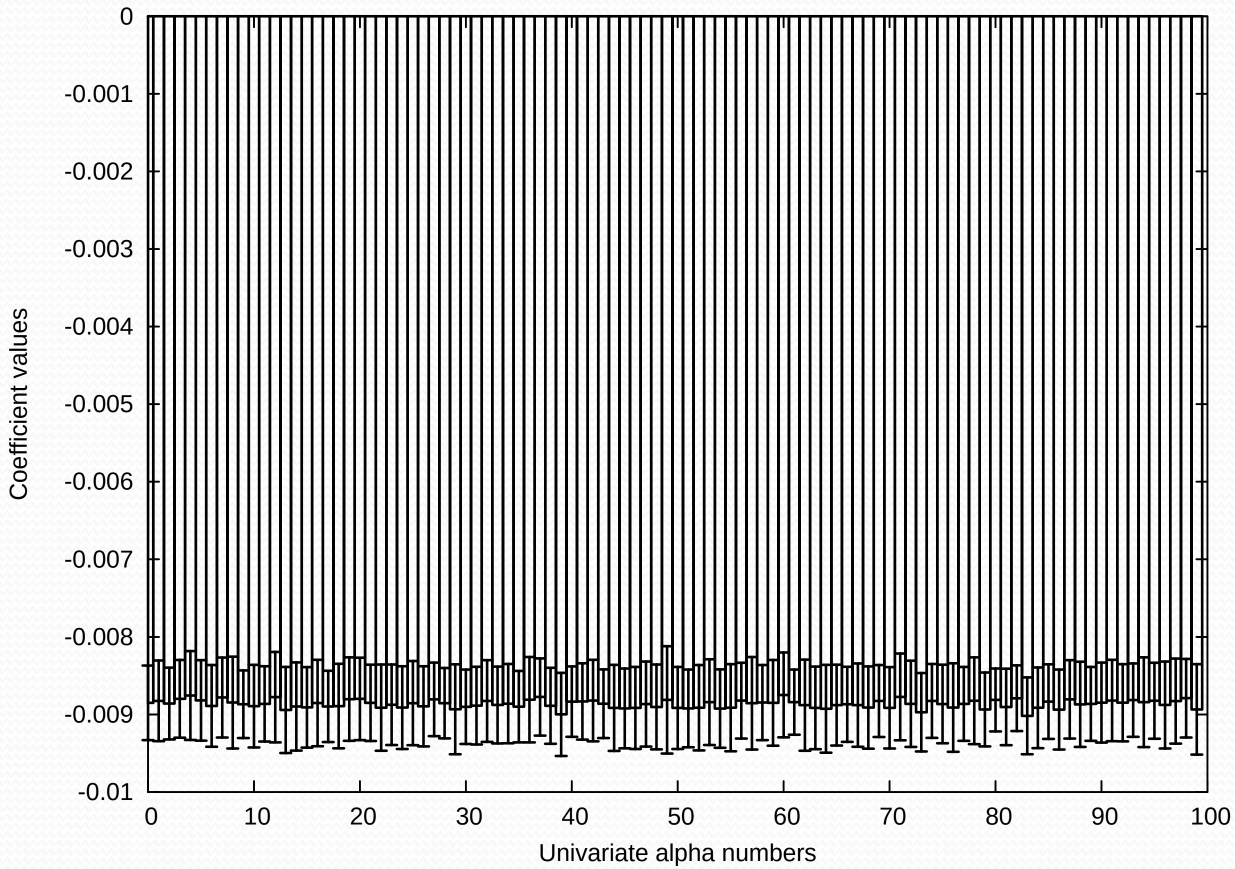
- Context: optimisation, meta-heuristics, evolutionary algorithms
- Fitness models, the MFM, and DEUM EDA
- Speedup – FM as surrogates
- Decision support – mining FM
- What makes a good model? – and the broader impact

Decision support

- MFM model coefficients
- The model points to solutions that are probably high in fitness
- $\alpha > 0$: bit should be 0, or bits should differ in value
- $\alpha < 0$: bit should be 1, or bits should be equal in value
- The following are models built using a single randomly generated population – values are mean from 100 runs

Model coefficients

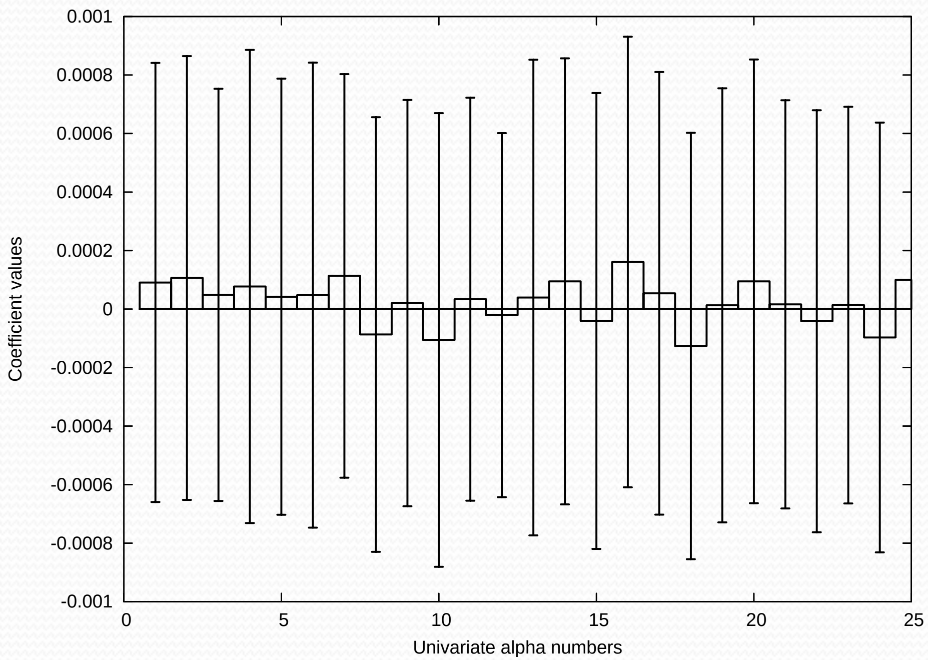
- Onemax
- Fitness is count of variables with value 1 (maximise)
- All variables have equal weight and should be 1

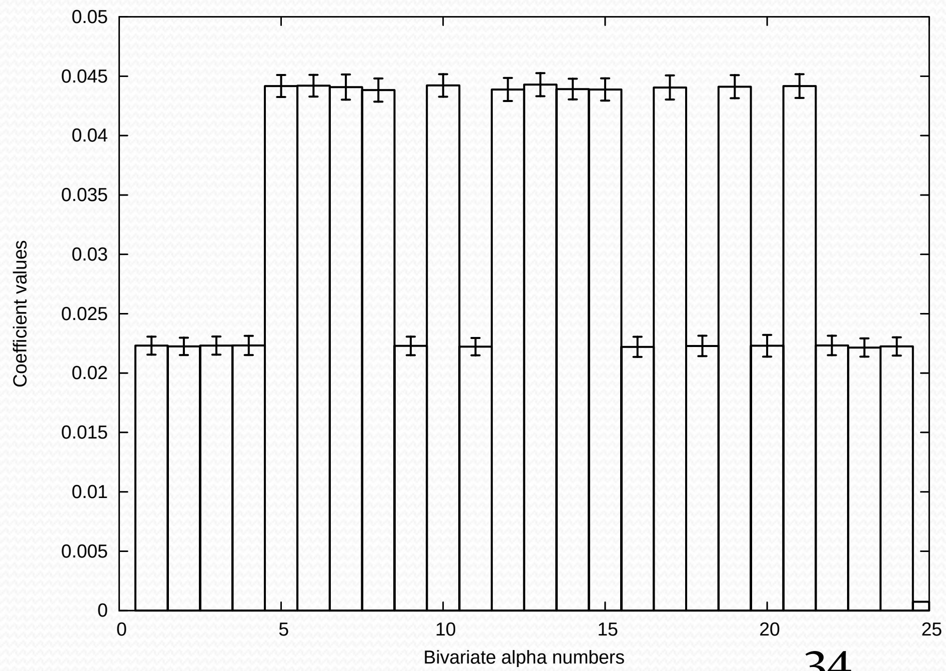


Model coefficients

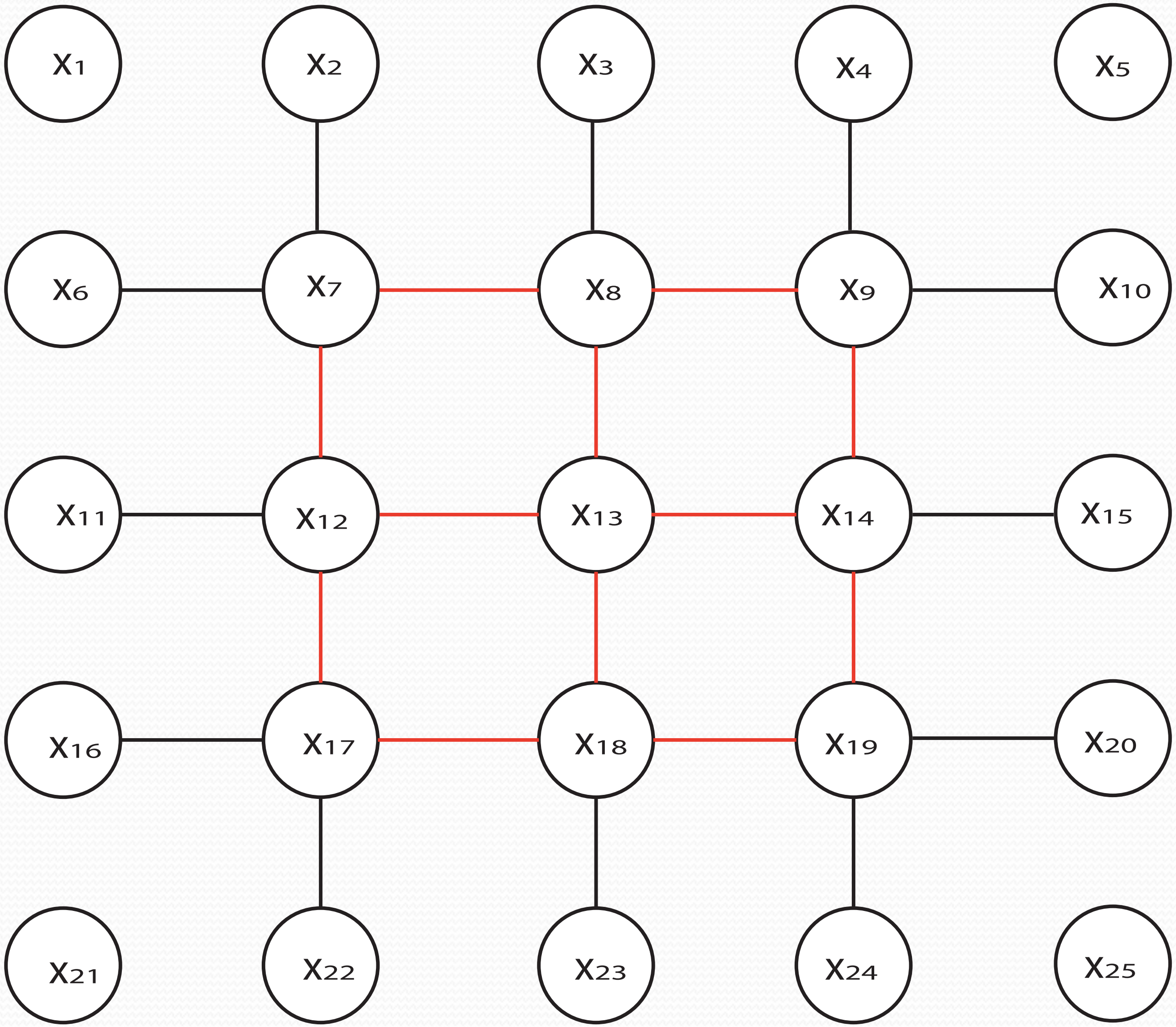
- 2D checkerboard
- Maximise neighbours that are different in value
- Includes pairwise interactions

X_0	X_1	X_2	X_3	X_4
X_5	X_6	X_7	X_8	X_9
X_{10}	X_{11}	X_{12}	X_{13}	X_{14}
X_{15}	X_{16}	X_{17}	X_{18}	X_{19}
X_{20}	X_{21}	X_{22}	X_{23}	X_{24}



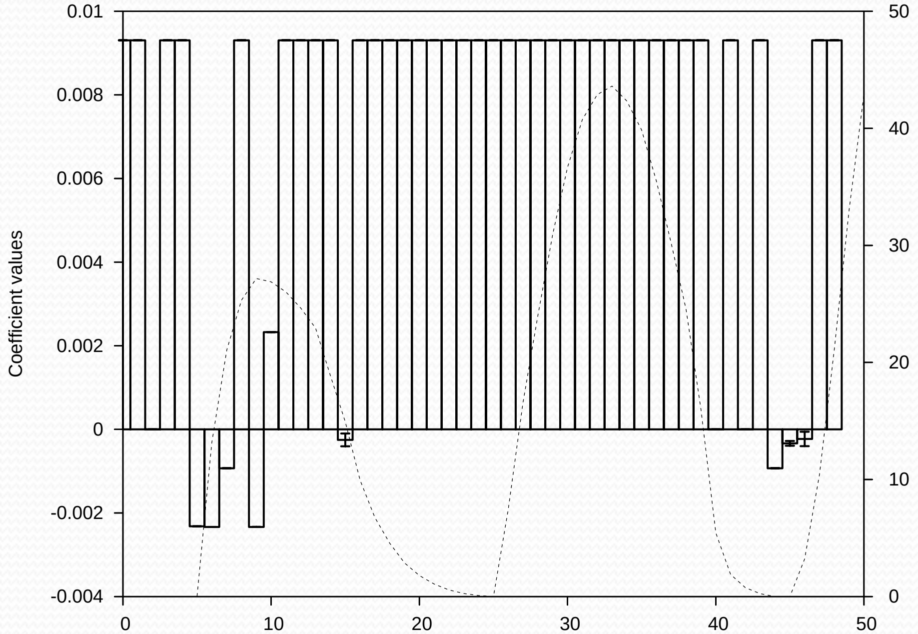


Model coefficients

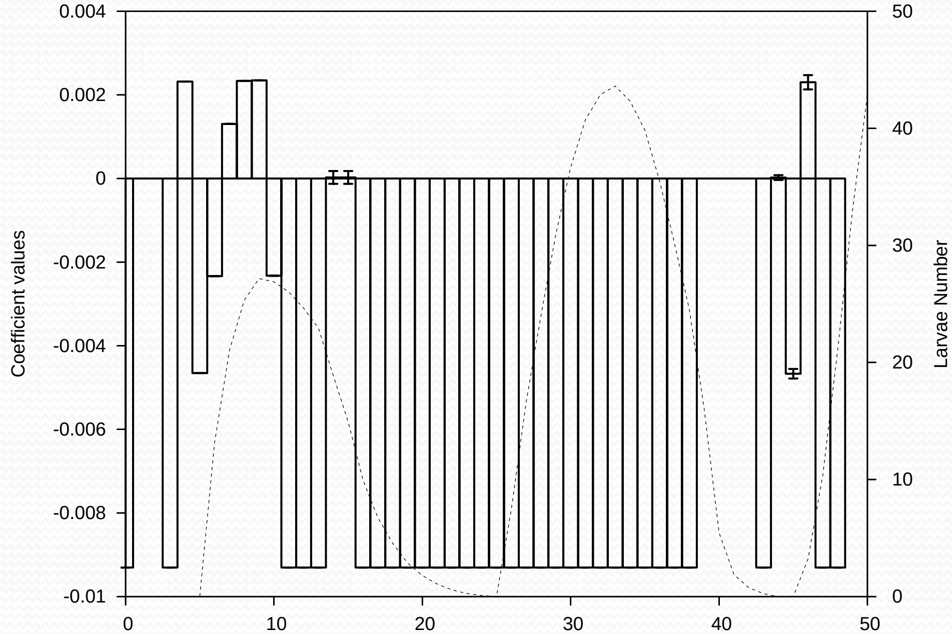


Model coefficients

- Bio-control (Mushrooms)
- Predicted intervention point match lifecycle of sciarid larvae



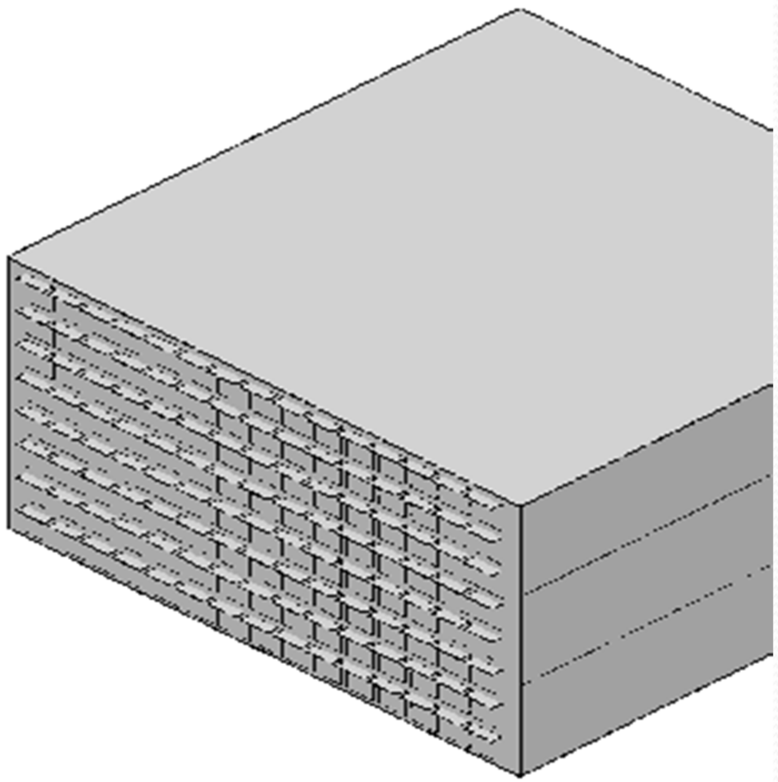
Univariate



Bivariate

Cellular Windows

- Ideal placement of glazing on a building façade; minimise energy use



0.017	0.017	0.016	0.016	0.015	0.015	0.015	0.015	0.016	0.015	0.015	0.015	0.016	0.016	0.016
0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.015	0.015	0.016	0.015	0.016	0.016	0.016
0.017	0.017	0.016	0.016	0.015	0.016	0.016	0.016	0.016	0.016	0.015	0.016	0.016	0.016	0.016
0.017	0.016	0.017	0.017	0.016	0.016	0.015	0.015	0.016	0.015	0.016	0.017	0.016	0.017	0.017
0.017	0.017	0.017	0.016	0.016	0.016	0.016	0.016	0.015	0.016	0.016	0.016	0.017	0.017	0.017
0.017	0.018	0.018	0.017	0.018	0.017	0.017	0.016	0.016	0.017	0.016	0.017	0.016	0.017	0.016
0.018	0.017	0.018	0.018	0.018	0.017	0.017	0.017	0.018	0.017	0.017	0.017	0.016	0.017	0.017
0.017	0.018	0.017	0.018	0.018	0.018	0.018	0.019	0.018	0.019	0.017	0.018	0.018	0.017	0.018

Outline

- Context: optimisation, meta-heuristics, evolutionary algorithms
- Fitness models, the MFM, and DEUM EDA
- Speedup – FM as surrogates
- Decision support – mining FM
- What makes a good model? – and the broader impact

What makes a good model?

- How do we define "good model"?
- Population size
- Structure (cliques in the model)
- Selection (filtering of solutions)

Fitness Prediction Correlation

- A measure of fitness prediction capability
- Procedure:
  - Construct fitness model
  - Generate population P
  - Predict fitnesses of P
  - Compute true fitnesses of P using fitness function
  - Calculate Spearman's rank correlation between predicted and true fitnesses

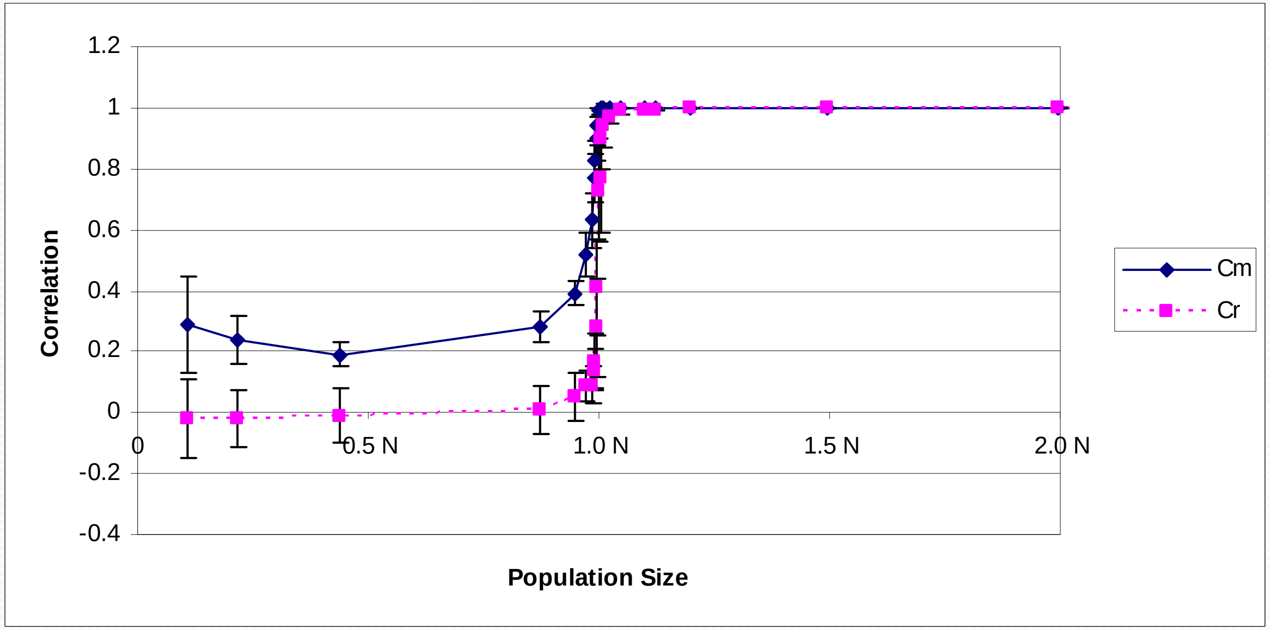
Two FPC figures

- C_r and C_m
- $\mathbf{C_r}$ is the FPC for a randomly generated population
- $\mathbf{C_m}$ is the FPC for a "neighbour population" – the solutions used to build the model each mutated 1 bit

- C_m is relevant because in one generation an EA moves between two similar populations

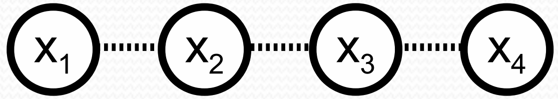
Population size

- Number of solutions used to build the model is important

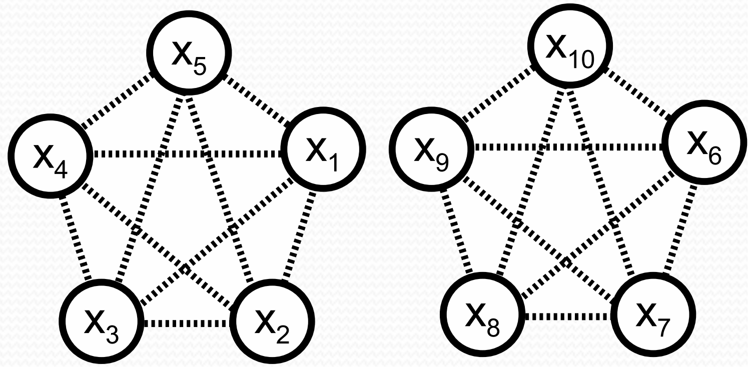


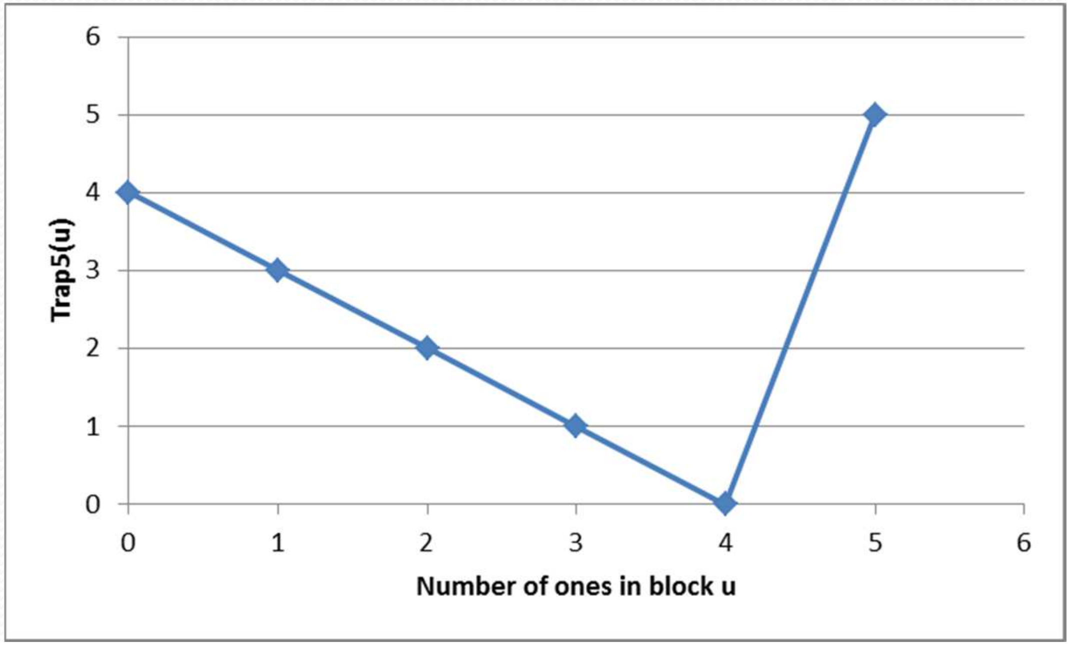
- Underspecified: population size < n
- Fully + over specified: population size >= n

Structure



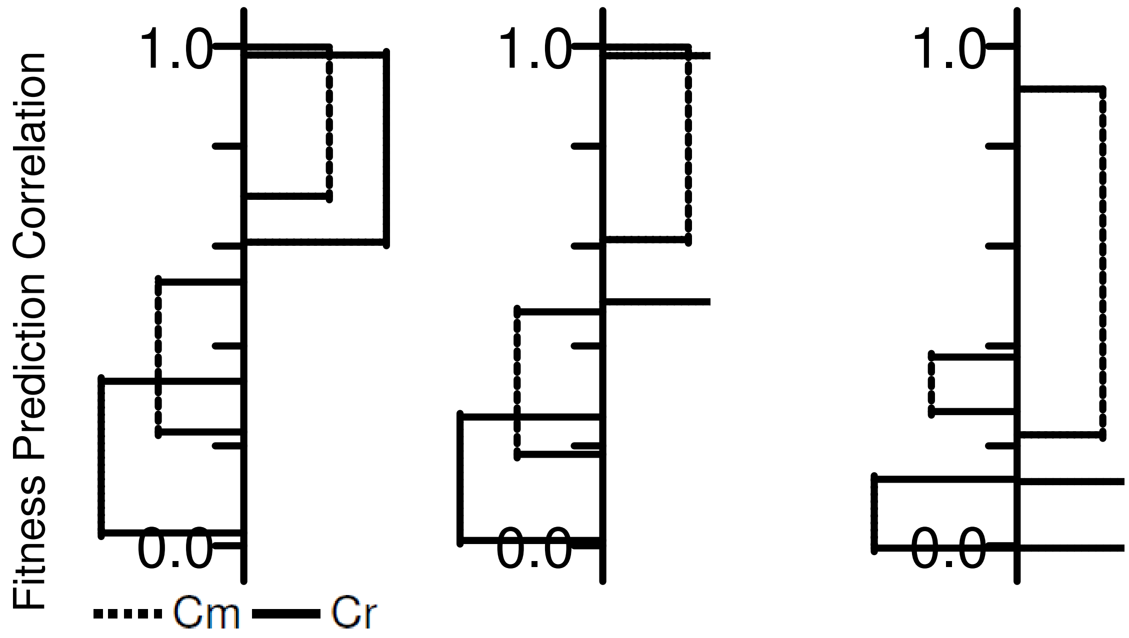
- 1D checkerboard
  - Optima 01010101.... and 10101010....
- Trap-5
  - Bitstring divided into groups of 5
  - Local optimum:
    - 000000....
  - Global optimum:
    - 111111111....





Structure

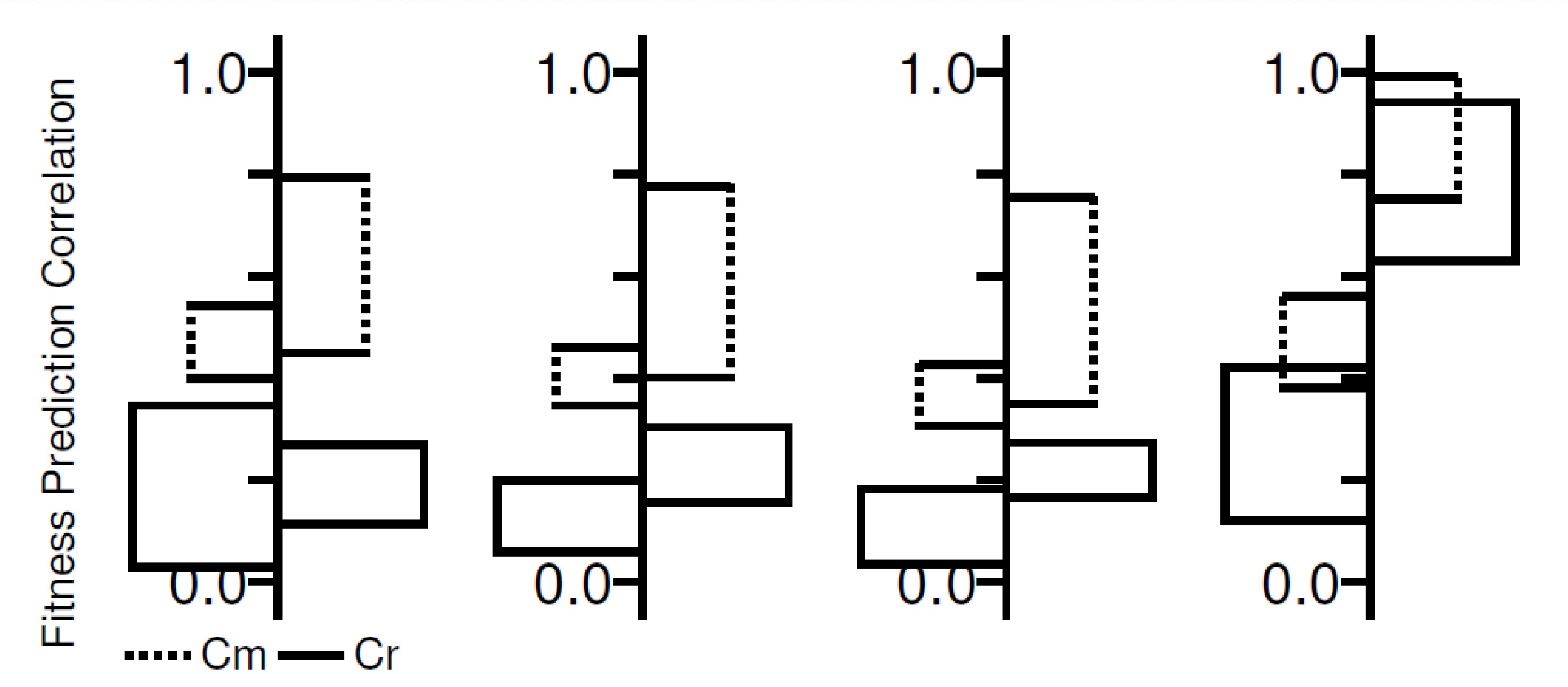
- 1D checkerboard
- Aggregated over instances from 10-1000 bits



1+2-cliques 2-cliques 1-cliques

Structure

- Trap-5
- Aggregated over instances with 20-100 bits



1-cliques 1+2 cliques 1+2+5 cliques 1+2+3+4+5 cliques

Structure

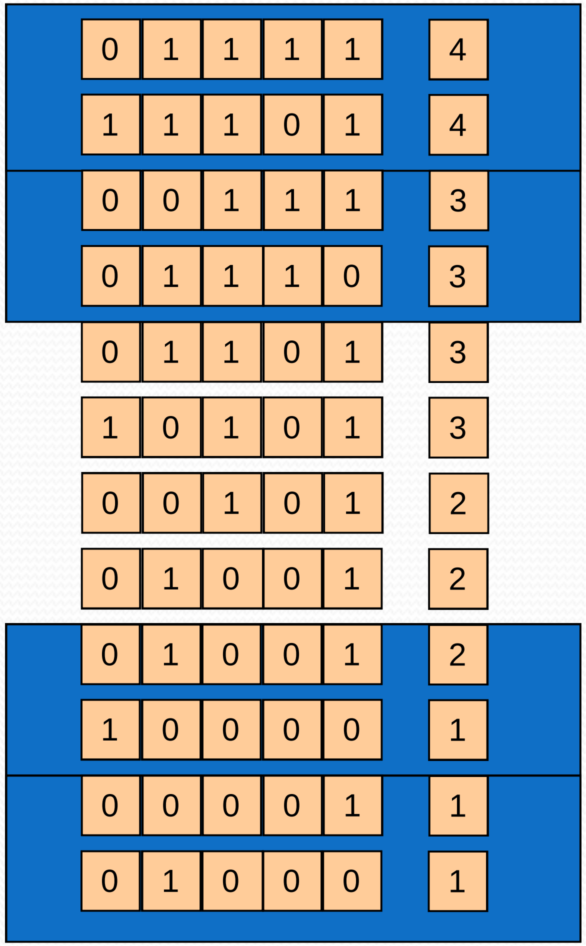
- Different parts of structure more / less important for good model (ranking solutions)
- Recent work: most problems have “structure”
- How much of it do we need to know about to determine ranking?
- Can we use knowledge of problem structure to map hard problems onto easier ones?

Selection

- Selection is not needed to choose population for estimating MFM parameters
  - But nothing to stop us using selection as a filter
- Many EDAs use truncation selection
  - Crude but inexpensive; selects top n individuals and discards the rest
  - This study looked at the impact of selection on fitness information within the population
- Many other selection operators exist, e.g.
  - Fitness proportionate (roulette wheel)
  - Tournament

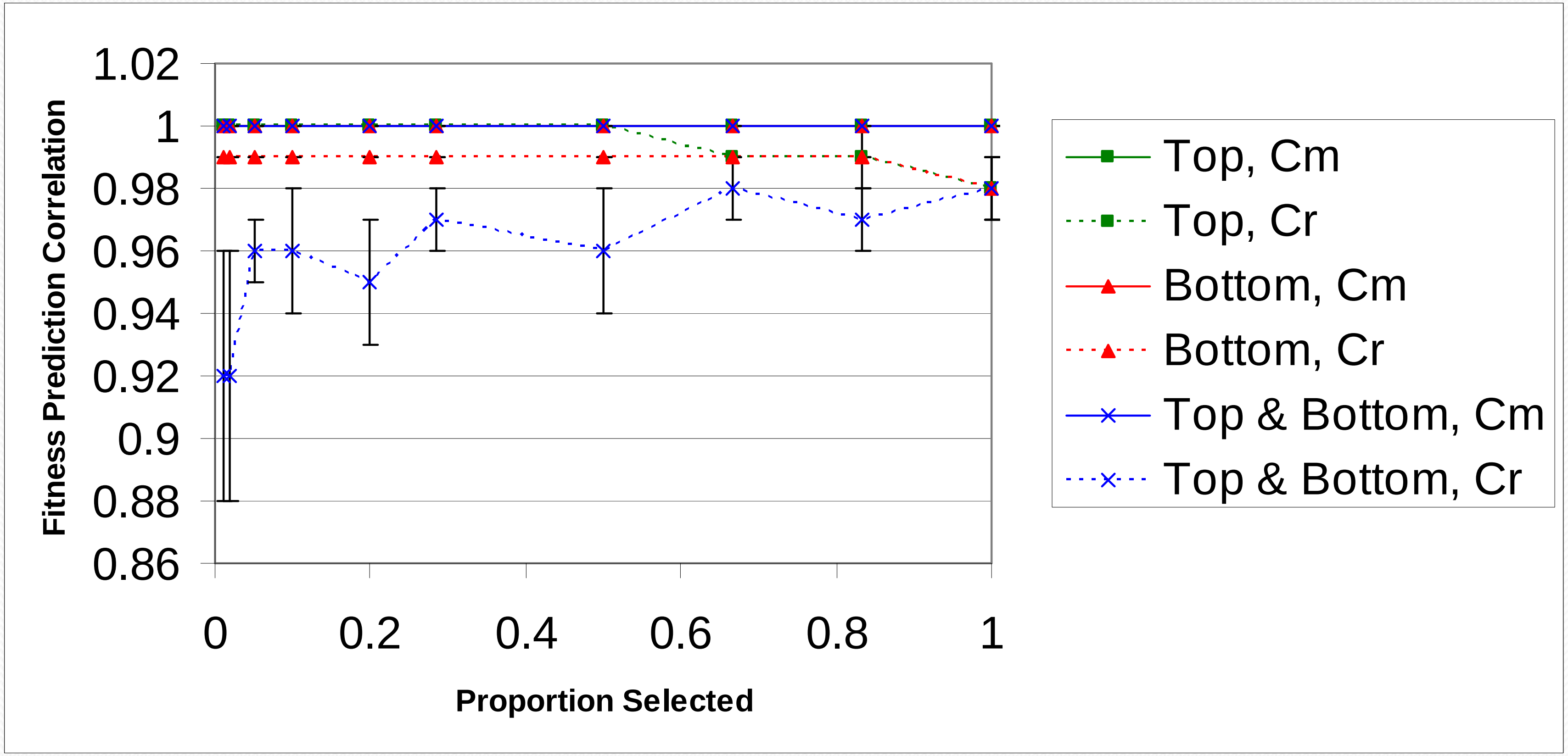
Selection operators

- Top selection
  - Standard Truncation
- Bottom selection
- Top & Bottom selection



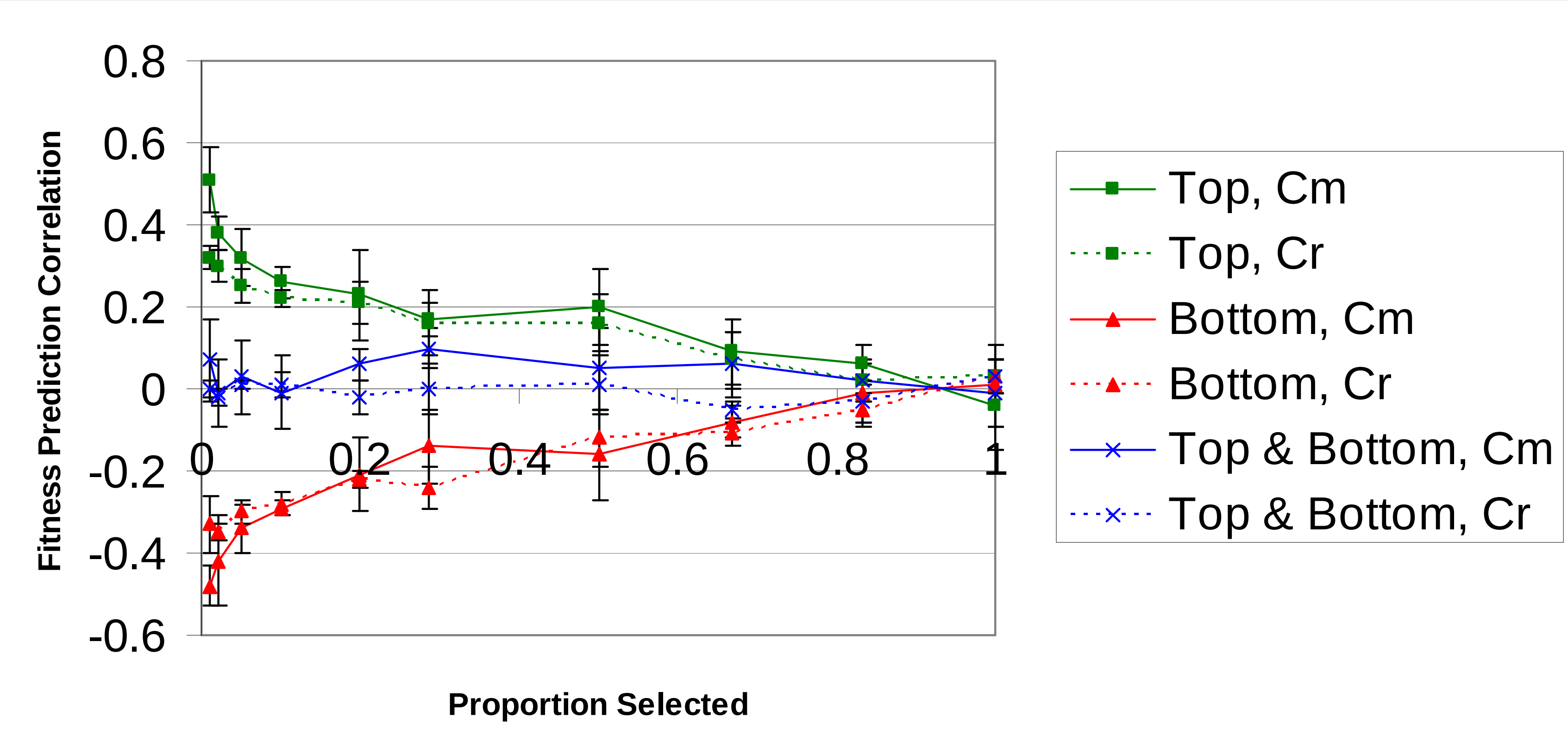
100 bit OneMax

- Fully specified, perfect model structure



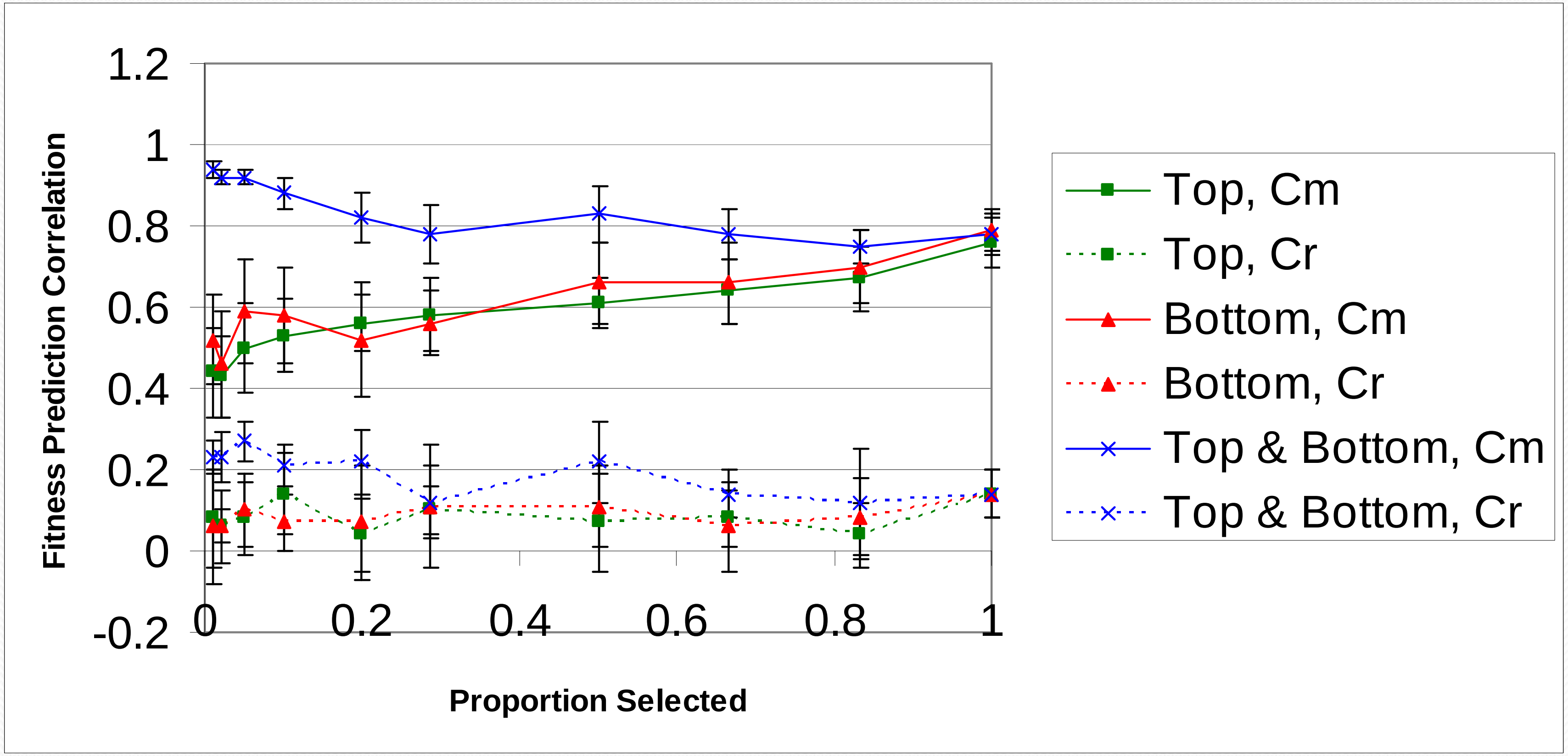
100 bit MaxSAT

- Underspecified, perfect model structure



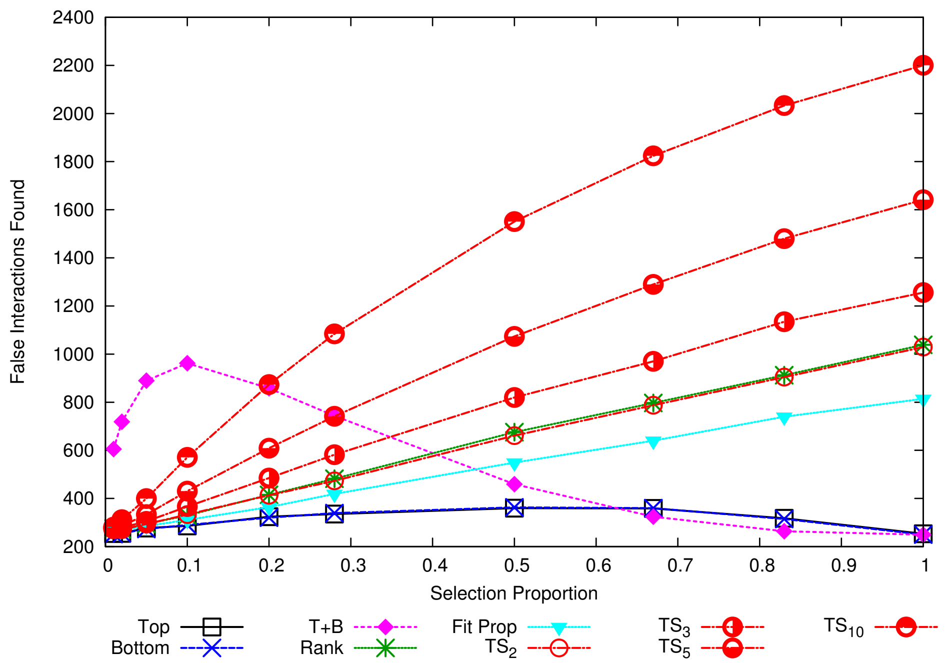
100 bit MaxSAT

- Fully specified, univariate model structure



Selection / structure learning

- Selection often part of structure learning
- Here, learning “structure” for onemax problem
- For most operators, spurious interactions increase with selection proportion
- Truncation selection most consistent
- Top / Bottom selection have similar results
- T+B worst for a low selection proportion



What makes a good model?

- With a perfect structure and big population, selection operator makes little difference (though top is still best)
- With small population or imperfect structure (more realistic), selection helps sharpen fitness information in population as well as providing pressure on search
- Thought: If we can build a good model, then there was useful information about fitness in the population
  - Results indicate that useful information exists in wider population
  - Can this be used in other algorithms?

Fitness models: summary

- Example fitness model: the MFM
- Useful for speeding up search
- Can be mined to aid decision making
- Can aid greater understanding of evolutionary operators

Where next?

- Which FM to use for a given problem?
- Modelling different spaces (e.g. permutations)
- Further exploration of selection's impact on search, model building and structure learning
- What do we mean by "structure" in a model, and in a problem, and how important is it anyway?
- Can we simplify optimisation problems – especially given knowledge of structure?

Thanks

- Question time
- Happy to discuss further – sbr@cs.stir.ac.uk
- Papers etc. at http://www.cs.stir.ac.uk/~sbr/