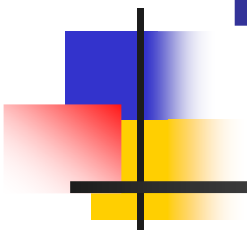


Simheuristics for the Multiobjective Nondeterministic Firefighter Problem in a Time-Constrained Setting

Krzysztof Michalak

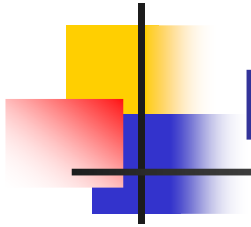
Wroclaw University of Economics, Poland

krzysztof.michalak@ue.wroc.pl

Joshua D. Knowles

University of Birmingham, UK

j.knowles.1@cs.bham.ac.uk



Presentation Plan

- The Firefighter Problem (FFP)
- Simheuristic approach
- The Sim-EA algorithm
- Experiments
- Results
- Conclusions

The Firefighter Problem (FFP)

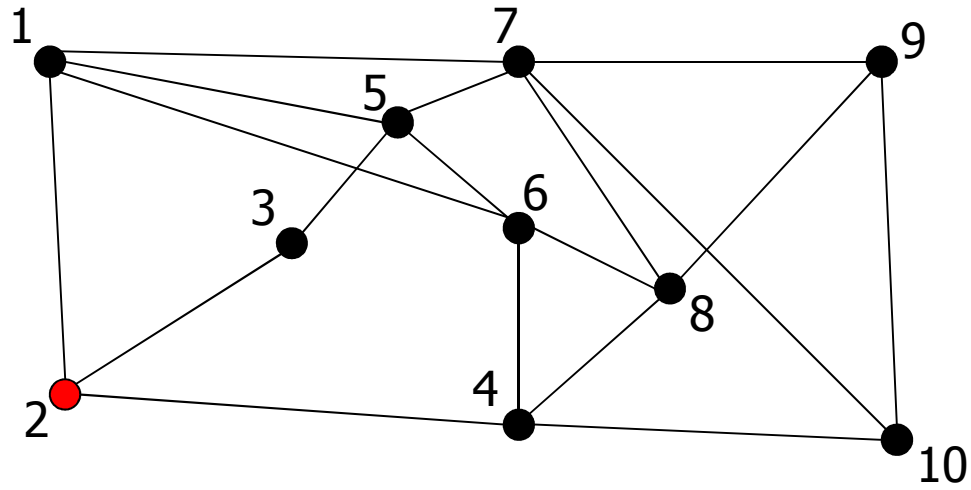
- Introduced by Hartnell in 1995[1]
- Spread of fire is modelled on an undirected graph
  - Discrete-time model
  - Vertices are labelled:
    - 'B' – burning
    - 'D' – defended by firefighters
    - 'U' – untouched
  - Initially a certain number N_s of nodes are burning ('B') and the remaining ones are untouched

[1] Hartnell, B.: *„Firefighter! An application of domination"*, in: 20th Conference on Numerical Mathematics and Computing (1995)

The Firefighter Problem (FFP)

- At each time step
  - N_f firefigters are assigned to the untouched ('U') nodes. These nodes become defended ('D')
  - The fire spreads: nodes adjacent to the burning nodes catch on fire (unless they are defended by firefighters)
- The simulation stops when the fire is contained or when all undefended nodes are burning
- Representation of a solution:
  - A permutation = the order in which the nodes are protected
  - Other representations possible

The Firefighter Problem – An Example



$N_s = 1$

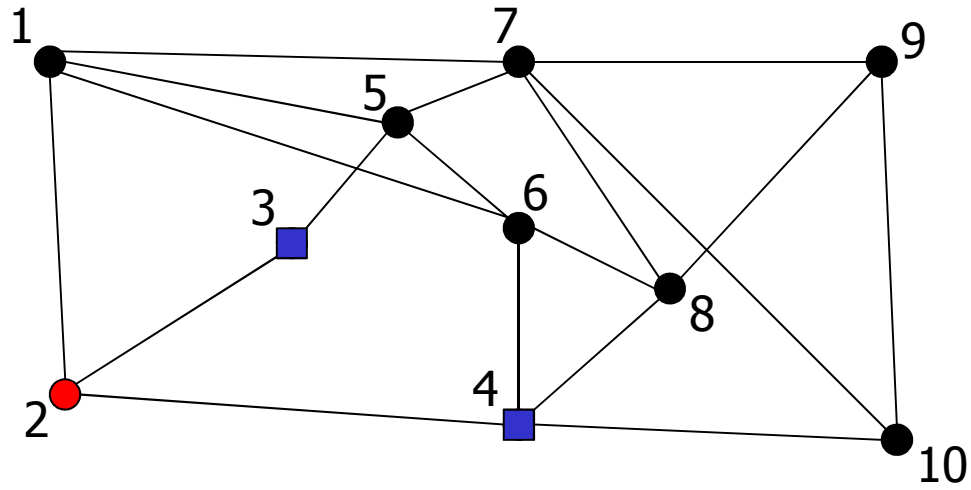
$N_f = 2$

P = 3, 4, 2, 1, 6, 7, 8, 9, 10, 5

$t = 0$

- N_s initial nodes are burning

The Firefighter Problem – An Example



$N_s = 1$

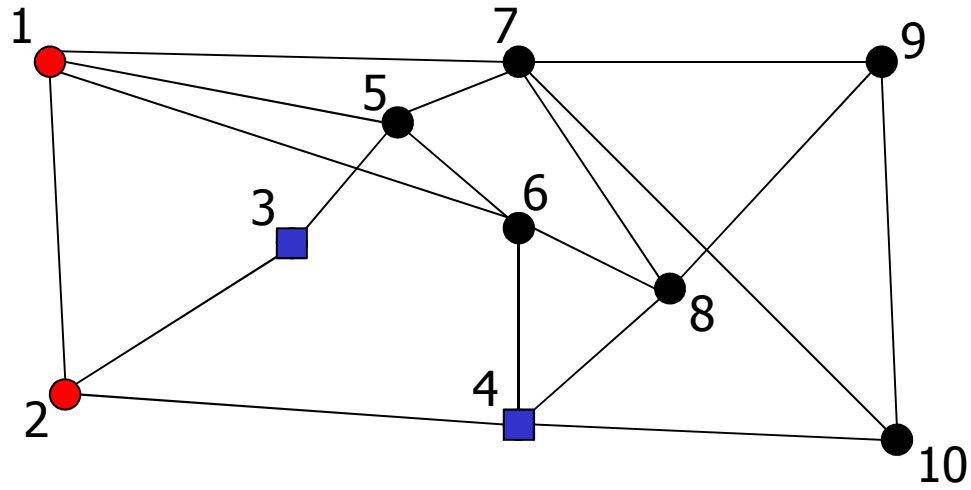
$N_f = 2$

P = 3, 4, 2, 1, 6, 7, 8, 9, 10, 5

$t = 1$

- N_f firefighters are assigned ■

The Firefighter Problem – An Example



$N_s = 1$

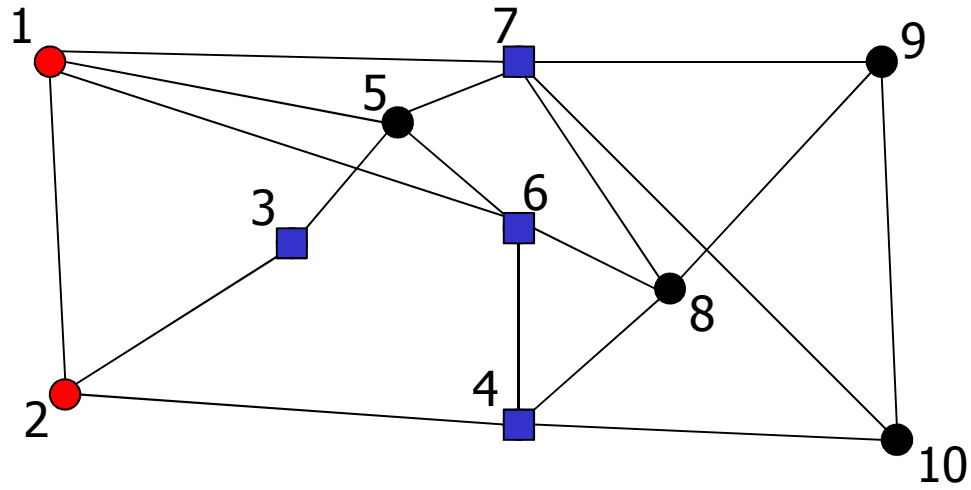
$N_f = 2$

P = 3, 4, 2, 1, 6, 7, 8, 9, 10, 5

$t = 1$

- the fire spreads to adjacent nodes

The Firefighter Problem – An Example



$N_s = 1$

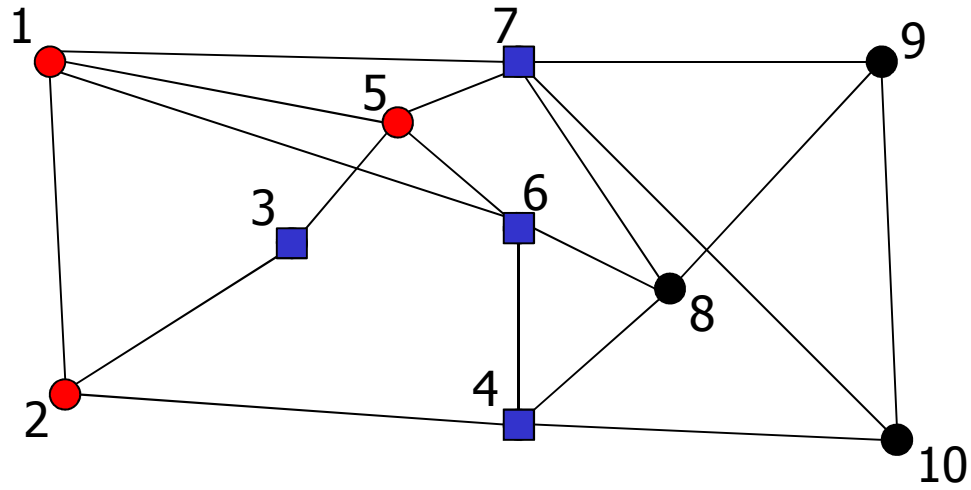
$N_f = 2$

P = 3, 4, 2, 1, 6, 7, 8, 9, 10, 5

$t = 2$

- N_f firefighters are assigned ■

The Firefighter Problem – An Example



$N_s = 1$

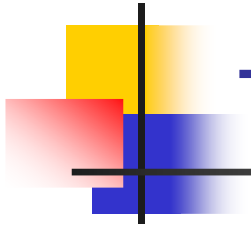
$N_f = 2$

P = 3, 4, 2, 1, 6, 7, 8, 9, 10, 5

$t = 2$

- the fire spreads to adjacent nodes ●

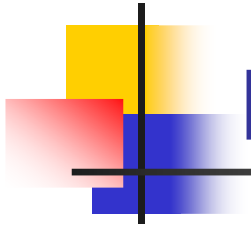
Note, that at this point the fire cannot spread any further and the simulation can be stopped



The Firefighter Problem (FFP)

- Task: Find the best assignment of firefighters to graph nodes
- Single-objecitve: save the highest possible number of nodes in the graph
- Multi-objective*): each node has several values assigned

*) **Introduced in:** K. Michalak *„Auto-adaptation of Genetic Operators for Multi-objective Optimization in the Firefighter Problem"*, Intelligent Data Engineering and Automated Learning IDEAL 2014, Lecture Notes in Computer Science, vol. 8669, pp. 484-491, Springer, 2014.



Nondeterministic FFP

- Deterministic FFP is a static optimization problem (given a permutation π we always get the same objectives)
- Nondeterministic FFP: fire spreads from node to (an adjacent) node with a probability $P_{spread} < 1$
- Nondeterministic FFP is a dynamic optimization problem
- Approaches to dynamic optimization
  - Offline: optimize once, then apply the solution
  - Online: modify the solution as fire spreads

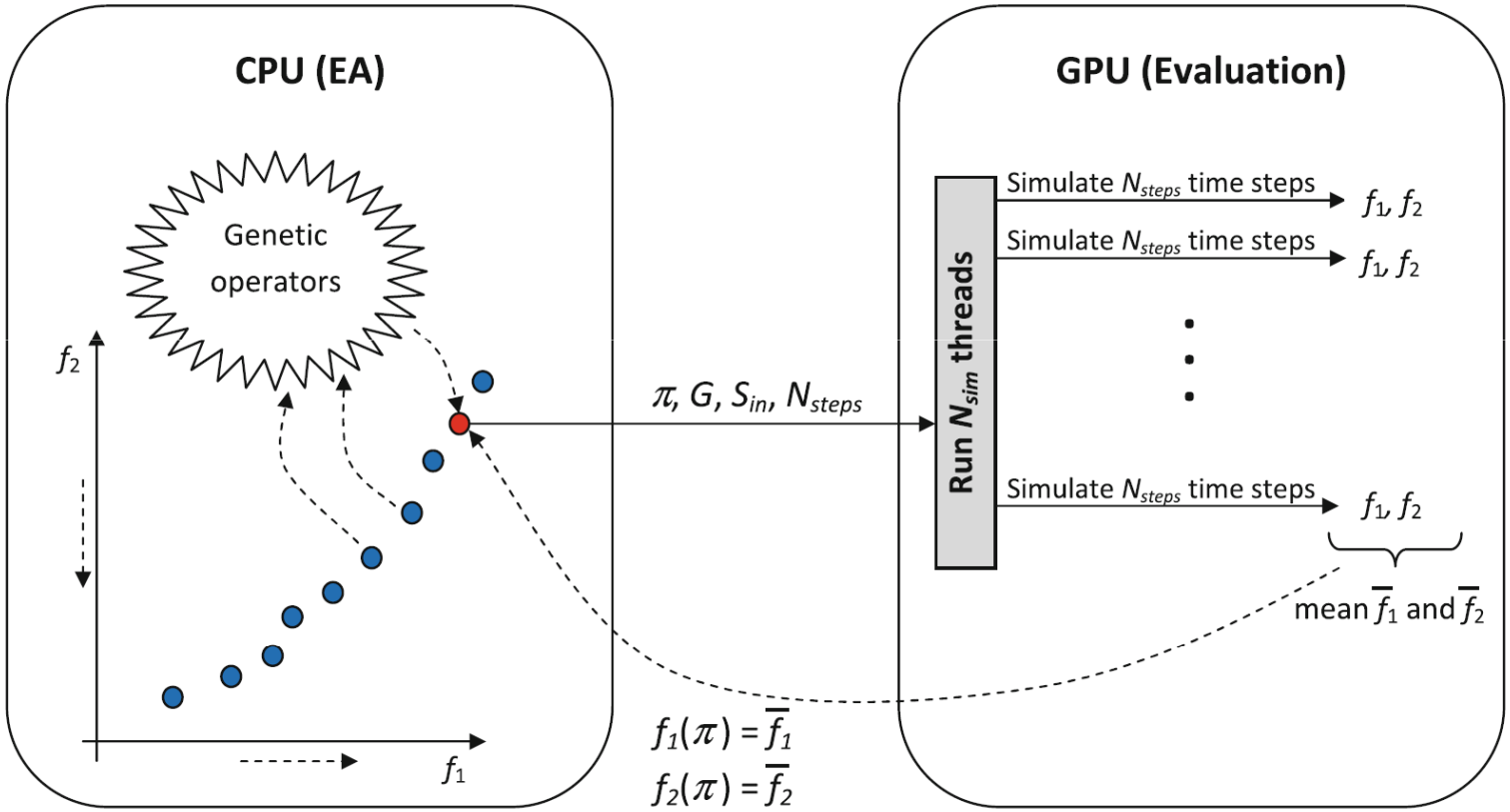
Simheuristic*) approach

- The main algorithm is a heuristic (e.g. EA)
- Spread of fire is simulated to evaluate solutions
- In the nondeterministic FFP each simulation run can give different results
- We perform N_{sim} = 200 simulations and average the results
- High computational cost

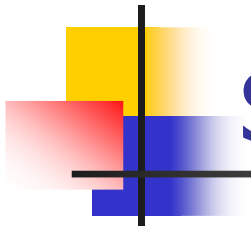
*) **A concept discussed in:** Juan, A.A.: *"A review of simheuristics: extending metaheuristics to deal with stochastic combinatorial optimization problems"*. Oper. Res. Perspect. 2, 62–72 (2015)

Simheuristic approach

- CPU+GPU architecture is used

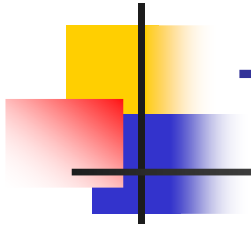


Implementation of the simheuristic on a CPU/GPU machine.



Simheuristic approach

- Question: how long to run the simulations?
- The number of simulated time steps: N_{steps}
- Smaller N_{steps}
  - More EA generations per time unit
  - Simulation results closer to the current state
- Larger N_{steps}
  - Less EA generations per time unit
  - Simulation results closer to the final state
- We tested N_{steps} = 2, 4, 6, 8, 10 and ∞ (unlimited, i.e. until the fire no longer spreads)



The Sim-EA algorithm

- A multipopulation algorithm…
- …but each subpopulation tackles a slightly different instance of the problem
- Island model + migration
- Migration influenced by similarities of problem instances
- Elitism (in each subpopulation separately)
- First applied to the TSP in

K. Michalak, *„Sim-EA: An Evolutionary Algorithm Based on Problem Similarity"*, Intelligent Data Engineering and Automated Learning - IDEAL 2014, Lecture Notes in Computer Science, vol. 8669, pp. 191-198, Springer, 2014.

The Sim-EA algorithm

Algorithm 3. The main loop of the Sim-EA algorithm (see also [12] and [13])

IN: $P_1, P_2, \ldots, P_{N_{sub}}$ - populations, one for each search direction $\lambda^{(d)}$
S_{in} - state of the graph for which to optimize
OUT: $P_1, P_2, \ldots, P_{N_{sub}}$ - populations after evolution

for $d := 1, \ldots, N_{sub}$ **do** // Initial evaluation
  Evaluate(S_{in}, P_d, $\lambda^{(d)}$)
end for
while not StoppingConditionMet() **do**
  for $d := 1, \ldots, N_{sub}$ **do** // Genetic operators
    P' := GeneticOperators(P_d)
    Evaluate(S_{in}, P', $\lambda^{(d)}$)
    $P_d := P_d \cup P'$
  end for

  for $d := 1, \ldots, N_{sub}$ **do** // Source populations
    s := SelectSourcePopulation(d)
    P'_d := the N_{imig} best specimens from P_s
  end for
  for $d := 1, \ldots, N_{sub}$ **do** // Migration
    for $x \in P'_d$ **do**
      Evaluate(S_{in}, $\{x\}$, $\lambda^{(d)}$)
      w := the weakest specimen in P_d
      $P_d := P_d - \{w\}$
      b := BinaryTournament($w, x, \lambda^{(d)}$)
      $P_d := P_d \cup \{b\}$
    end for
  end for

  for $d := 1, \ldots, N_{sub}$ **do** // Elitist selection
    e := the best specimen in P_d
    P_d := Select($P_d \backslash \{e\}$, $N_{pop} - 1$)
    $P_d := P_d \cup \{e\}$
  end for
end while

Proceedings, Part 2
page 254

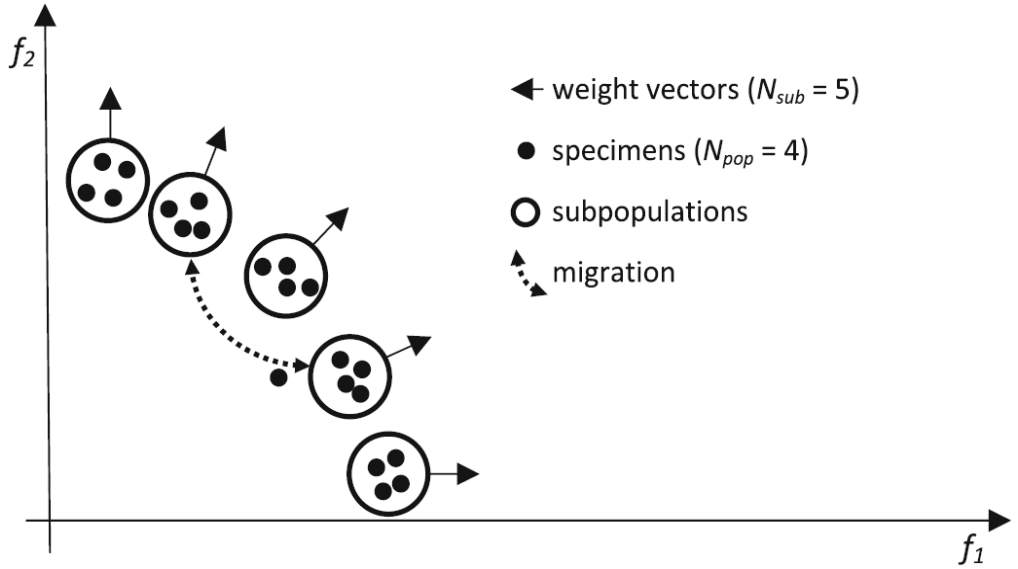
select immigrants

perform migration

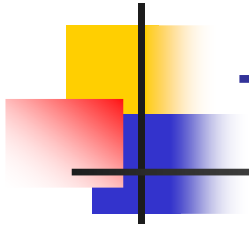
selection with elitism

The Sim-EA for Multiobjective Optimization

- Decomposition-based approach
- Each subpopulation tackles a subproblem with a different weight vector
- Problem similarity = a dot product of their weight vectors



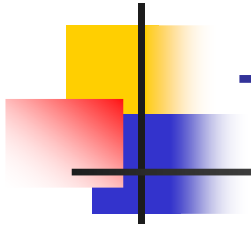
An overview of elements of the Sim-EA algorithm.



The Sim-EA for the Multiobjective FFP

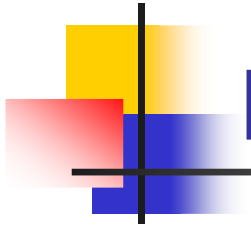
- Motivation
  - To be able to tackle a nondeterministic version of the problem
  - Dynamic optimization
  - Online interaction with the developing fire
  - We need to simulate each trade-off between the objectives separately
- Last year's paper on the deterministic FFP

  K. Michalak *„The Sim-EA Algorithm with Operator Autoadaptation for the Multiobjective Firefighter Problem"*, 15th European Conference, EvoCOP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings, Lecture Notes in Computer Science, volume 9026, str. 184–196, Springer, 2015.
- A stepping stone towards this work



The Sim-EA for the Multiobjective FFP

- Conclusions from the 2015 paper:
  - The Sim-EA can utilize computational resources more effectively than MOEA/D
  - The best migration scheme: rank
    - Rank all subopulations from the largest dot product to the smallest
    - Apply roulette-wheel selection
  - Migration significantly improves the results obtained by the Sim-EA algorithm

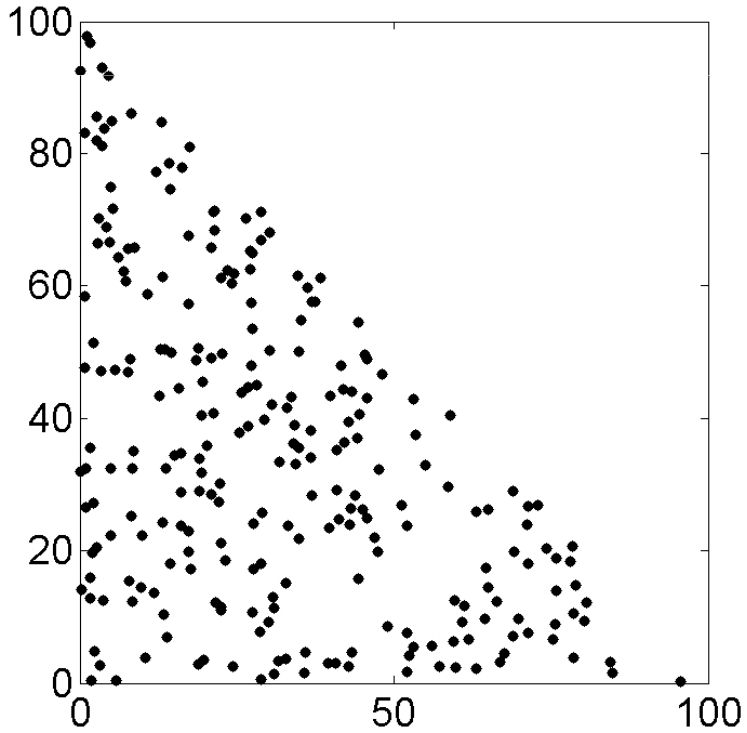


Experiments

- Investigated issues:
  - Comparing the online and offline optimization
  - Comparing the optimizers with some simple heuristics
  - Determining the influence of the N_{steps} parameter on the quality of the obtained results
- Stopping criterion – a time limit
  - Offline optimizer: 300 second total
  - Online optimizer: 60 seconds per time step

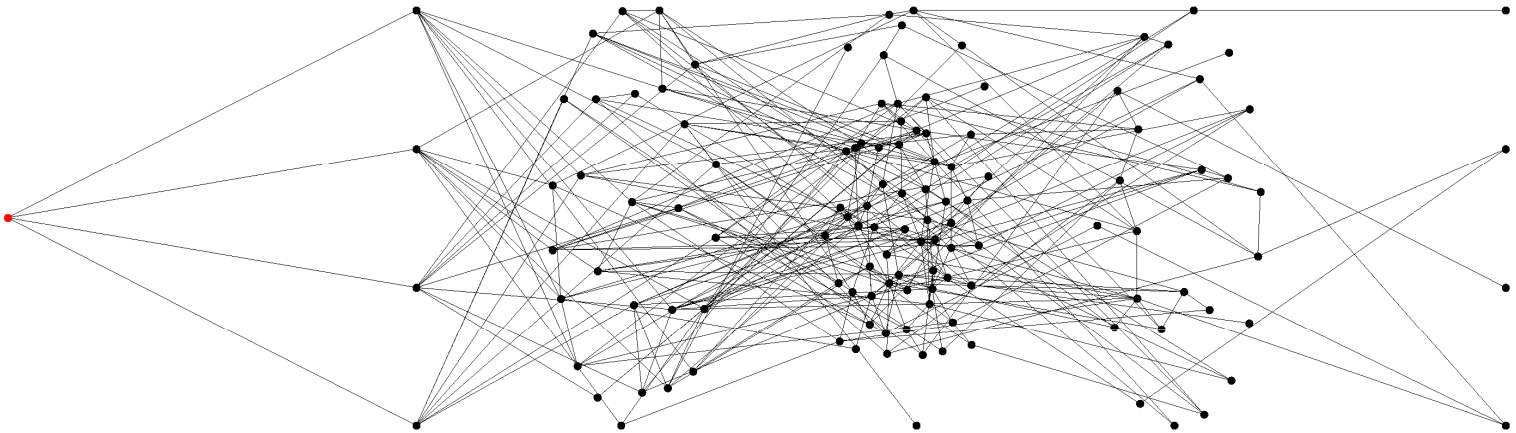
Experiments

- Test data = graphs with N_v = 30, 40, 50, 75, 100 and 125
- For each pair of vertices an edge is added with a probability $P_{edge} = 2.5 / N_v$
- $N_s = 1$, $N_f = 2$
- Cost assignment
  - Pairs of random values with a uniform probability on a triangle formed by points [0, 0], [0, 100], [100, 0]
  - Sum of costs associated with a vertex cannot exceed 100
  - It is not possible to maximize both objectives at the same time



Experiments

- An example of a graph with $N_v = 125$ vertices



- 30 test instances for each N_v
- Probability of fire spreading: $P_{sp} = 0.3$, 0.5, 0.7, 0.9 and 1.0
- $N_{sub} = 20$ subpopulations of $N_{pop} = 100$ specimens each
- Number of immigrants: $N_{imig} = 0.1 \cdot N_{pop} = 10$

Results – offline vs. online optimizer

- 30 measurements obtained for the best value of N_{steps} for each of the optimizers
- For smaller instances (N ≤ 50) both optimization approaches are more or less comparable
- For larger instances (N ≥ 75) the online optimizer outperformed the offline one 12 times (with 3 cases undecided, none in favour of the offline optimizer)

The results of a statistical comparison of the online and offline optimization modes: 'N' - online better, 'F' - offline better, '=' - no statistical difference.

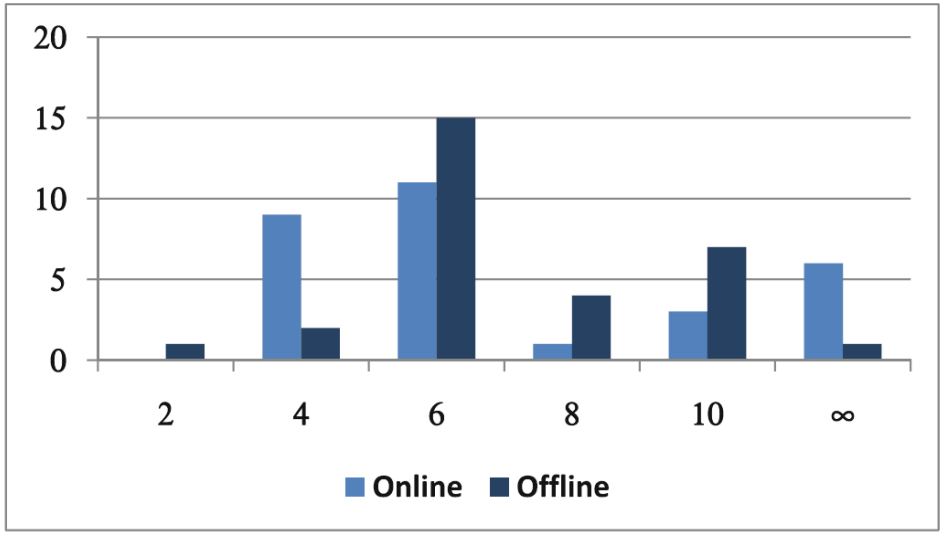
N_v	$P_{sp} = 0.3$	$P_{sp} = 0.5$	$P_{sp} = 0.7$	$P_{sp} = 0.9$	$P_{sp} = 1.0$
30	=	N	N	=	=
40	N	=	=	=	=
50	=	N	=	=	F
75	=	N	N	N	=
100	N	N	N	N	=
125	N	N	N	N	N

Results – simulation length N_{steps}

- Letting the simulation run without limit can deteriorate the results of the optimization (cf. the figure)
- For larger instances (N_v = 75, 100 and 125):
  - The unlimited simulation length worked best for $P_{sp} \leq 0.5$
  - For $P_{sp} \geq 0.7$ limiting the duration of simulations worked better
  - See Tables 1-6 in the paper

Hypothesis: For lower values of P_{sp}:

- harder to predict the short-time movement of the fire
- averaged behaviour observed in the longer run becomes more important



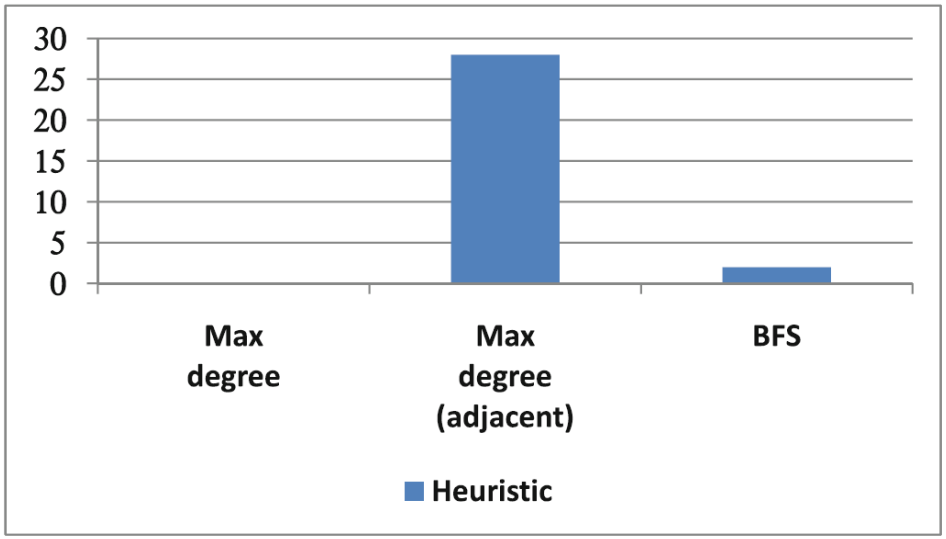
The number of times each value of N_{steps} produced the best result.

Results – optimization vs. heuristics

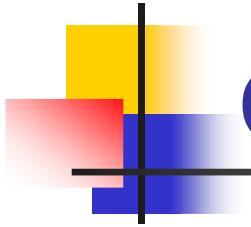
- Three heuristics tested
- Best: select the vertex with highest degree adjacent to the fire
- Comparison with the optimizers:
  - $P_{sp} \leq 0.5$ – results in favour of the heuristics
  - $P_{sp} \geq 0.7$ – results in favour of the optimizers

Hypothesis: For lower values of P_{sp}:

- harder to predict the short-time movement of the fire
- heuristics may be more effective than simulations with a high level of uncertainty



The number of times each heuristic produced the best result.



Conclusions

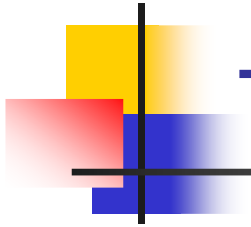
- The nondeterministic firefighter problem was studied
- This problem is a Dynamic Optimization Problem (DOP)
- The Sim-EA algorithm allowed us to simulate different trade-offs between objectives simultaneously
- Online optimization is comparable (small instances) or better (larger instances) than offline optimization
- Hypothesis: for low P_{sp} it is hard to obtain good evaluations of solutions using simulations
  - Unlimited simulation length worked best for $P_{sp} \leq 0.5$
  - The "Max degree (adjacent)" heuristic is very effective when $P_{sp} \leq 0.5$
- More work on hybrids is warranted

Thank you!

(questions?)

Additional Slides

Sim-EA



The Sim-EA algorithm

- A multipopulation algorithm…
- …but each subpopulation tackles a slightly different instance of the problem
- Island model + migration
- Migration influenced by similarities of problem instances
- Elitism (in each subpopulation separately)
- First applied to the TSP in

K. Michalak, *„Sim-EA: An Evolutionary Algorithm Based on Problem Similarity"*, Intelligent Data Engineering and Automated Learning - IDEAL 2014, Lecture Notes in Computer Science, vol. 8669, pp. 191-198, Springer, 2014.

The Sim-EA algorithm

Algorithm 3. The main loop of the Sim-EA algorithm (see also [12] and [13])

IN: $P_1, P_2, \ldots, P_{N_{sub}}$ - populations, one for each search direction $\lambda^{(d)}$
S_{in} - state of the graph for which to optimize
OUT: $P_1, P_2, \ldots, P_{N_{sub}}$ - populations after evolution

for $d := 1, \ldots, N_{sub}$ **do** // Initial evaluation
  Evaluate(S_{in}, P_d, $\lambda^{(d)}$)
end for
while not StoppingConditionMet() **do**
  for $d := 1, \ldots, N_{sub}$ **do** // Genetic operators
    P' := GeneticOperators(P_d)
    Evaluate(S_{in}, P', $\lambda^{(d)}$)
    $P_d := P_d \cup P'$
  end for

  for $d := 1, \ldots, N_{sub}$ **do** // Source populations
    s := SelectSourcePopulation(d)
    P'_d := the N_{imig} best specimens from P_s
  end for
  for $d := 1, \ldots, N_{sub}$ **do** // Migration
    for $x \in P'_d$ **do**
      Evaluate(S_{in}, $\{x\}$, $\lambda^{(d)}$)
      w := the weakest specimen in P_d
      $P_d := P_d - \{w\}$
      b := BinaryTournament($w, x, \lambda^{(d)}$)
      $P_d := P_d \cup \{b\}$
    end for
  end for

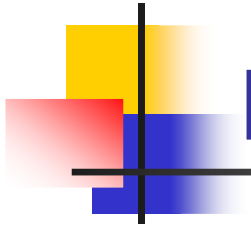
  for $d := 1, \ldots, N_{sub}$ **do** // Elitist selection
    e := the best specimen in P_d
    P_d := Select($P_d \backslash \{e\}$, $N_{pop} - 1$)
    $P_d := P_d \cup \{e\}$
  end for
end while

Proceedings, Part 2
page 254

select immigrants

perform migration

selection with elitism



Problem similarity

- Problem instance similarity matrix

$$S_{[N_{prob} \times N_{prob}]}$$

where:

N_{prob} – the number of problem instances

higher values = more similar instances

- Similarity between the *i*-th and *j*-th problem instances is expressed numerically as $S_{i,j}$

Problem similarity for the TSP

- In 2014 paper problem instances were represented as weighted graphs

- The *i* -th problem instance had a cost matrix:

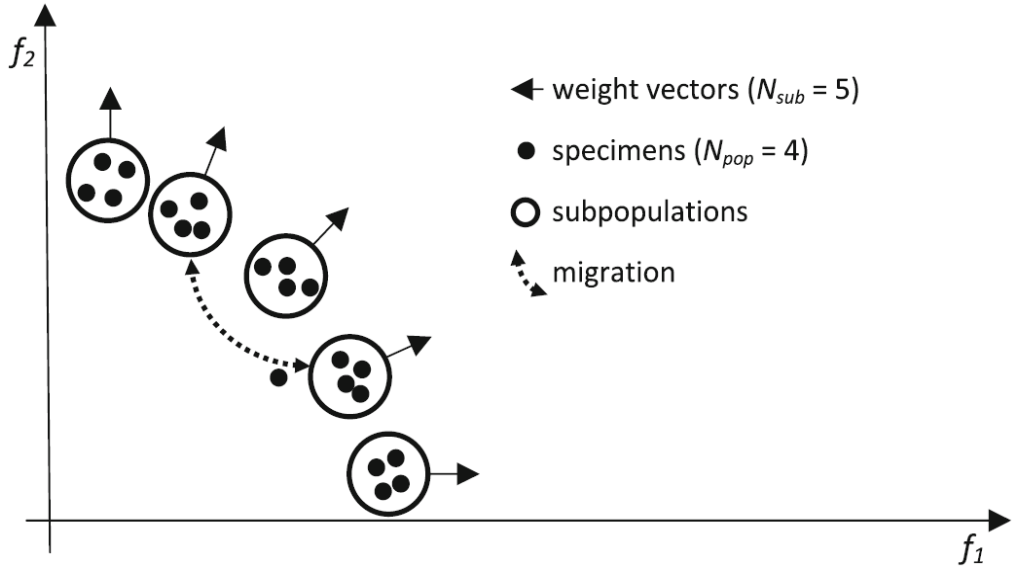
$$C^{(i)}_{[N \times N]}$$

- Similarity between *i* -th and *j* -th problem instance:

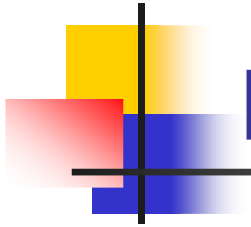
$$S_{i,j} = -\sum_{p=1}^{N}\sum_{q=1}^{N}(C^{(i)}_{p,q} - C^{(j)}_{p,q})^2.$$

The Sim-EA for Multiobjective Optimization

- Decomposition-based approach
- Each subpopulation tackles a subproblem with a different weight vector
- Problem similarity = a dot product of their weight vectors

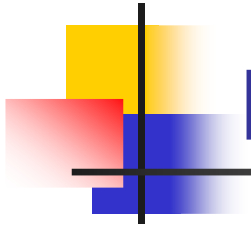


An overview of elements of the Sim-EA algorithm.



Migration

- For each destination population i select the source population j based on values of $S_{i,j}$
- Migration stategies
  - Nearest – migrate specimens from the most similar subproblem (largest $S_{i,j}$ for $j \neq i$)
  - Rank
    - Rank all subopulations from the largest dot product to the smallest
    - Apply roulette-wheel selection
  - Uniform – migrate specimens from a subproblem chosen at random with uniform probability (from $j \neq i$)
  - None – no migration



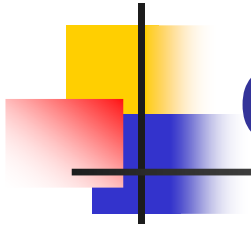
Migration

- Incoming specimens are added one by one
  - x – incoming specimen
  - w – the weakest specimen in target population
  - binary tournament between x and w
  - if x wins it replaces w

Operator Autoadaptation

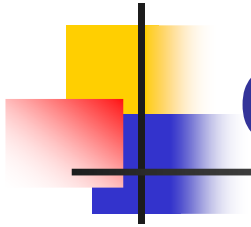
Operators

- Typical operators used for permutation-based problems
- 10 crossover operators
  - Cycle Crossover (CX)
  - Linear Order Crossover (LOX)
  - Merging Crossover (MOX)
  - Non-Wrapping Order Crossover (NWOX)
  - Order Based Crossover (OBX)
  - Order Crossover (OX)
  - Position Based Crossover (PBX)
  - Partially Mapped Crossover (PMX)
  - Precedence Preservative Crossover (PPX)
  - Uniform Partially Mapped Crossover (UPMX).



Operators

- 5 mutation operators
  - displacement mutation
  - insertion mutation
  - inversion mutation
  - scramble mutation
  - transpose mutation

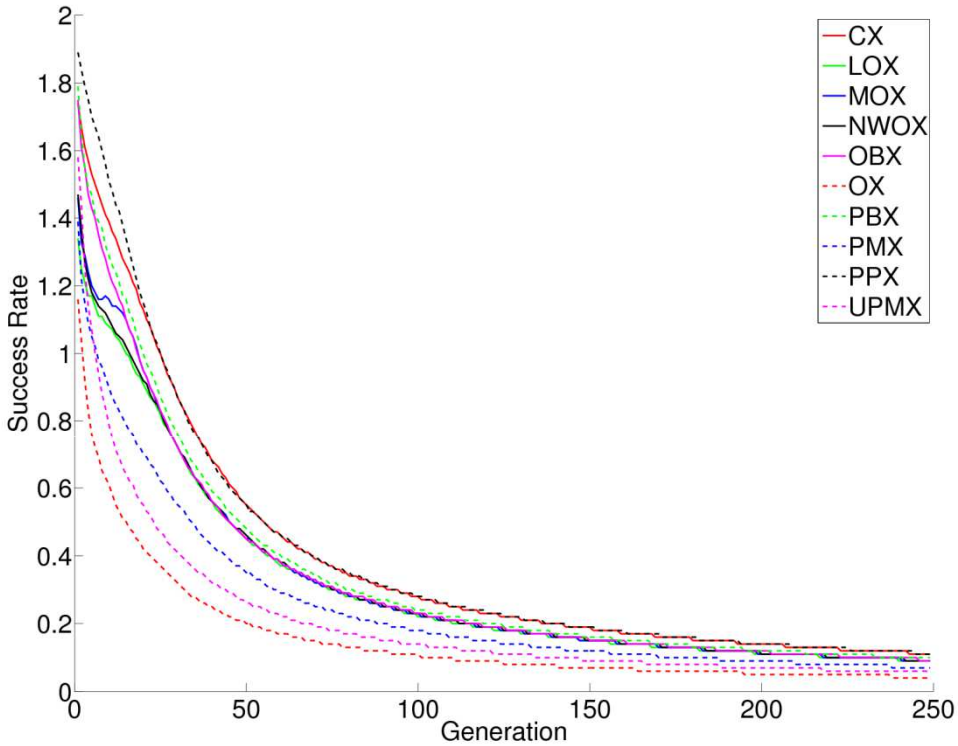


Operator Autoadaptation

- Based on:
  - b_i - the number of times when a given operator produced an improved specimen according to the aggregated objective
  - n_i - the number of times each operator was used
  - success rate $s_i = b_i / n_i$
- Probability assignment
  - Each of the N_{op} operators is given a minimum probability P_{min}
  - The remaining $1 - N_{op} \cdot P_{min}$ is divided proportionally to s_i
- The selection of crossover and mutation operators is performed separately

Autoadaptation (2015 paper)
Performance of the operators

- Crossover
  - best: CX and PPX
  - worst: OX

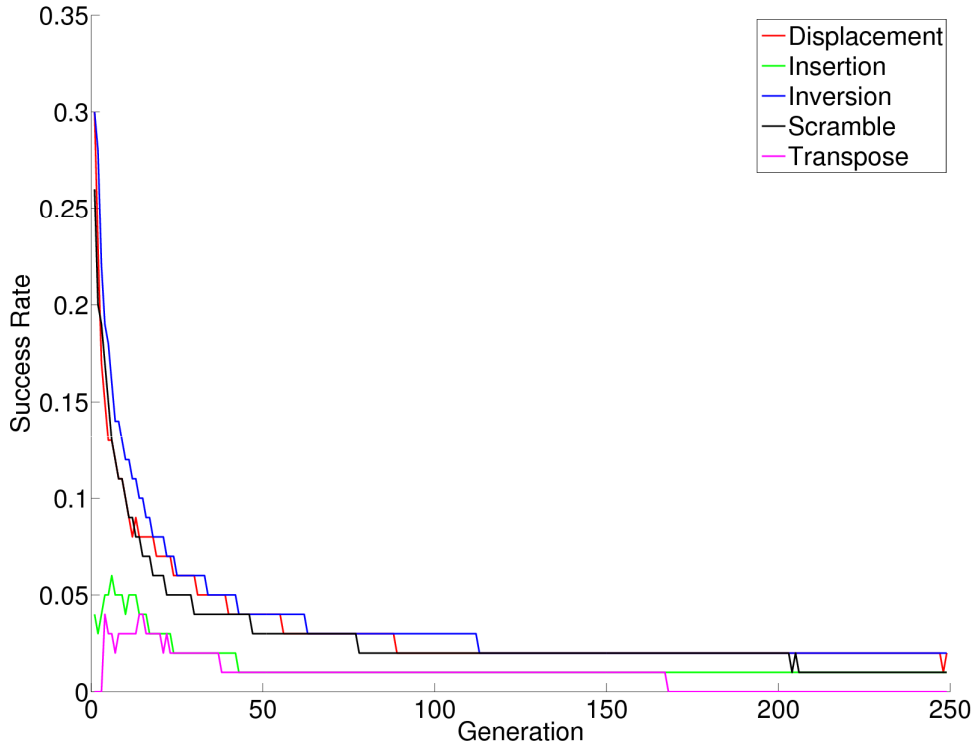


Success rates for crossover operators (Sim-EA, rank-based migration, N_v = 150)

K. Michalak *„The Sim-EA Algorithm with Operator Autoadaptation for the Multiobjective Firefighter Problem"*, 15th European Conference, EvoCOP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings, Lecture Notes in Computer Science, volume 9026, str. 184–196, Springer, 2015.

Autoadaptation (2015 paper)
Performance of the operators

- Mutation
  - best: displacement
  - slightly worse: scramble and inversion
  - worst: transpose



Success rates for crossover operators (Sim-EA, rank-based migration, N_v = 150)

K. Michalak *„The Sim-EA Algorithm with Operator Autoadaptation for the Multiobjective Firefighter Problem"*, 15th European Conference, EvoCOP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings, Lecture Notes in Computer Science, volume 9026, str. 184–196, Springer, 2015.