Simple Population Replacement Strategies for a Steady-State Multi-Objective Evolutionary Algorithm

Christine L. Mumford

christine@cs.cardiff.ac.uk

Cardiff University

Introduction

- We explore some simple evolutionary strategies

Introduction

- We explore some simple evolutionary strategies
- For the *SEAMO* algorithm (a simple evolutionary algorithm for multi-objective optimization)

Introduction

- We explore some simple evolutionary strategies
- For the *SEAMO* algorithm (a simple evolutionary algorithm for multi-objective optimization)
- SEAMO is a simple, elitist, steady-state Pareto-based evolutionary algorithm

Introduction

- We explore some simple evolutionary strategies
- For the *SEAMO* algorithm (a simple evolutionary algorithm for multi-objective optimization)
- SEAMO is a simple, elitist, steady-state Pareto-based evolutionary algorithm
- That uses simple rules for replacing individuals in the population instead of global fitness based on dominance ranking

The Objectives of the Study

- To discover the best replacement strategies

The Objectives of the Study

- To discover the best replacement strategies
- Then use them to improve SEAMO

Test problems

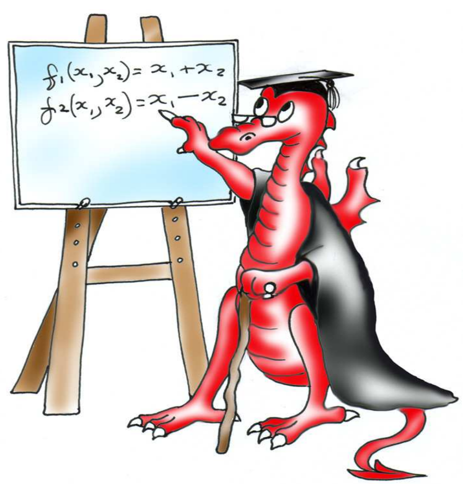
Test problems

- Multiple knapsack problems (MKPs)

Test problems

- Multiple knapsack problems (MKPs)

- Continuous functions, SPH-2, ZDT6, QV and KUR



The SEAMO Framework

- SEAMO sequentially selects every individual in the population, in turn, serve as the first parent

The SEAMO Framework

- SEAMO sequentially selects every individual in the population, in turn, serve as the first parent
- And pairs it with a second parent that is selected at random (uniformly)

The SEAMO Framework

- SEAMO sequentially selects every individual in the population, in turn, serve as the first parent
- And pairs it with a second parent that is selected at random (uniformly)
- A single crossover is then applied to produce one offspring

The SEAMO Framework

- SEAMO sequentially selects every individual in the population, in turn, serve as the first parent
- And pairs it with a second parent that is selected at random (uniformly)
- A single crossover is then applied to produce one offspring
- And this is followed by a single mutation

The SEAMO Framework

- SEAMO sequentially selects every individual in the population, in turn, serve as the first parent
- And pairs it with a second parent that is selected at random (uniformly)
- A single crossover is then applied to produce one offspring
- And this is followed by a single mutation
- Each new offspring will either replace an existing population member or it will die

SEAMO Pseudocode

Procedure *SEAMO*
Begin
Generate N random individuals {N is the population size}
Evaluate the objective vector for each population member and store it
Repeat
For each member of the population
This individual becomes the first parent
Select a second parent at random
Apply crossover to produce single offspring
Apply a single mutation to the offspring
Evaluate the objective vector produced by the offspring
if offspring qualifies
Then the offspring replaces a member of the population
else it dies
Endfor
Until stopping condition satisfied
Print all non-dominated solutions in the final population
End

Replacement Strategy for the Original SEAMO

Evaluation criteria:

Replacement Strategy for the Original SEAMO

Evaluation criteria:

1. Does offspring dominate either parent?

Replacement Strategy for the Original SEAMO

Evaluation criteria:

1. Does offspring dominate either parent?
2. Does offspring produce any global improvements to any Pareto components?

Replacement Strategy for the Original SEAMO

Evaluation criteria:

1. Does offspring dominate either parent?
2. Does offspring produce any global improvements to any Pareto components?

- On the basis of this superiority test, an offspring will replace one or other of its parents if it is deemed to be better

Replacement Strategy for the Original SEAMO

Evaluation criteria:

1. Does offspring dominate either parent?
2. Does offspring produce any global improvements to any Pareto components?

- On the basis of this superiority test, an offspring will replace one or other of its parents if it is deemed to be better
- Otherwise it will die

Replacement Strategy for the Original SEAMO

Evaluation criteria:

1. Does offspring dominate either parent?
2. Does offspring produce any global improvements to any Pareto components?

- On the basis of this superiority test, an offspring will replace one or other of its parents if it is deemed to be better
- Otherwise it will die
- Phenotypic duplicates are deleted, regardless

Detailed Replacement Strategy for Original SEAMO

1. **if** offspring dominates parent 1 (and it is not a duplicate), it replaces it

Detailed Replacement Strategy for Original SEAMO

1. **if** offspring dominates parent 1 (and it is not a duplicate), it replaces it
2. **else if** offspring dominates parent 2 (and it is not a duplicate), it replaces it

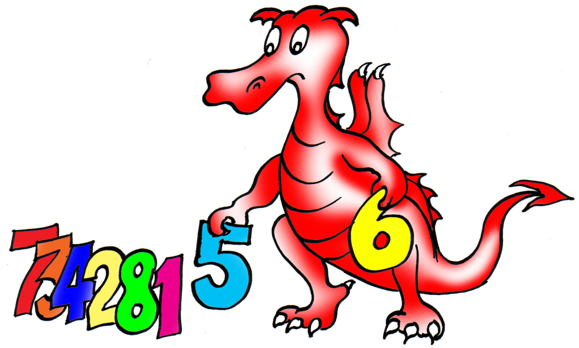
Detailed Replacement Strategy for Original SEAMO

1. **if** offspring dominates parent 1 (and it is not a duplicate), it replaces it
2. **else if** offspring dominates parent 2 (and it is not a duplicate), it replaces it
3. **else if** offspring harbors in new best-so-far Pareto component
   (a) it replaces a parent, provided no other best-so-far Pareto component is lost
   (b) occasionally, offspring will replace a random population member to avoid such a loss

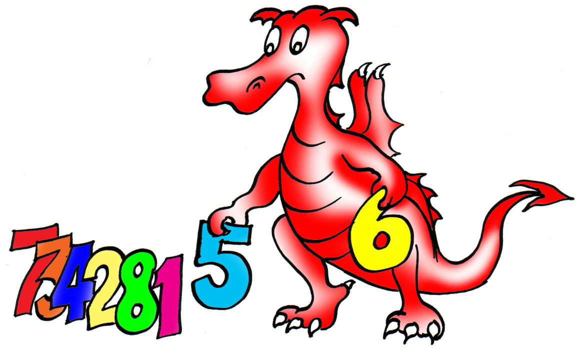
Detailed Replacement Strategy for Original SEAMO

1. **if** offspring dominates parent 1 (and it is not a duplicate), it replaces it
2. **else if** offspring dominates parent 2 (and it is not a duplicate), it replaces it
3. **else if** offspring harbors in new best-so-far Pareto component
   (a) it replaces a parent, provided no other best-so-far Pareto component is lost
   (b) occasionally, offspring will replace a random population member to avoid such a loss
4. **otherwise** it dies

Representation for the MKP

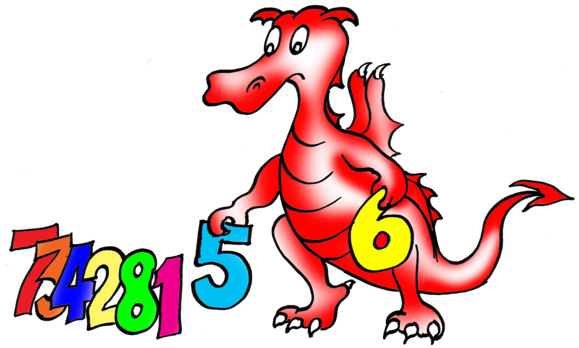


Representation for the MKP



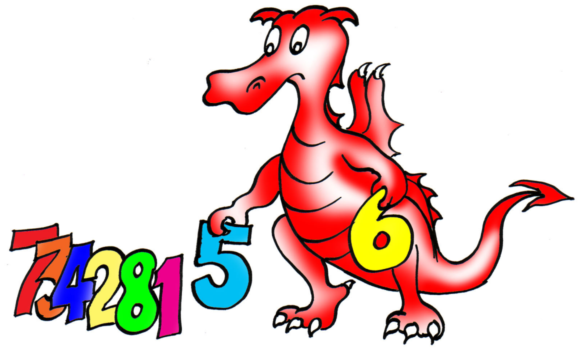
- Order-based representation with a first fit decoder

Representation for the MKP



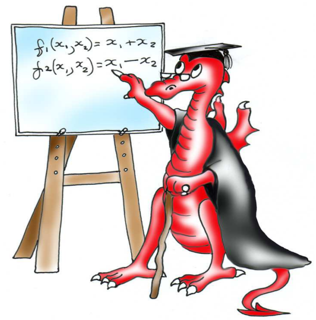
- Order-based representation with a first fit decoder
- Cycle Crossover (CX)

Representation for the MKP

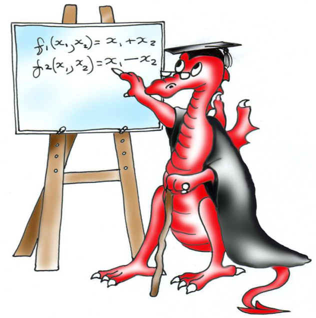


- Order-based representation with a first fit decoder
- Cycle Crossover (CX)
- A simple mutation operator swaps two arbitrarily selected objects within a single permutation list

The Continuous Functions

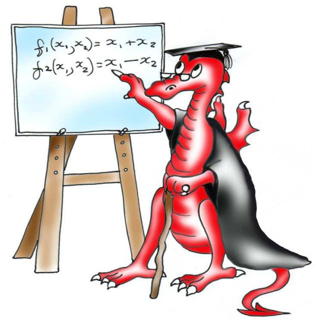


The Continuous Functions



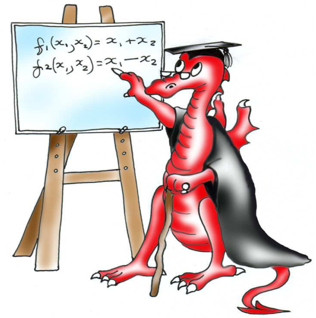
- Solutions are coded as real vectors of length 100

The Continuous Functions



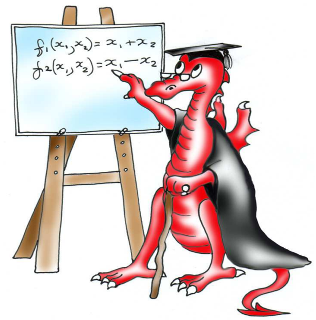
- Solutions are coded as real vectors of length 100
- One-point crossover

The Continuous Functions



- Solutions are coded as real vectors of length 100
- One-point crossover
- A non-uniform mutation

The Continuous Functions



- Solutions are coded as real vectors of length 100
- One-point crossover
- A non-uniform mutation
- Deletion of duplicates: component objective functions x_i and x'_i of $\mathbf{x}$ and $\mathbf{x}'$, are equal if and only if

  $x_i - \epsilon \leq x'_i \leq x_i + \epsilon$,
  where ϵ is an error term ($0.00001 \times x_i$)

Experimental Method

Experimental Method

- Each strategy is tested by 30 replicate runs, each initialized with a different random seed

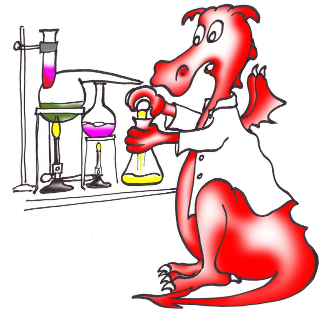
Experimental Method

- Each strategy is tested by 30 replicate runs, each initialized with a different random seed
- 2D plots are made by combining all the non-dominated solutions from all the 30 replicate run

Experimental Method

- Each strategy is tested by 30 replicate runs, each initialized with a different random seed
- 2D plots are made by combining all the non-dominated solutions from all the 30 replicate run
- 2D plots give a good comparisons for solutions quality, spread and range

Experimental Method



- Each strategy is tested by 30 replicate runs, each initialized with a different random seed
- 2D plots are made by combining all the non-dominated solutions from all the 30 replicate run
- 2D plots give a good comparisons for solutions quality, spread and range
- Performance metrics compare average performance of SEAMO with other EAs

Simple Replacement Strategies

1. offspring replaces a population member that it dominates at random

Simple Replacement Strategies

1. offspring replaces a population member that it dominates at random
2. offspring replaces a parent that it dominates

Simple Replacement Strategies

1. offspring replaces a population member that it dominates at random
2. offspring replaces a parent that it dominates
3. offspring replaces a parent if it dominates either parent, otherwise it replaces a population member that it dominates at random

Replacing a Population Member at Random

Repeat

Select population member at random without replacement

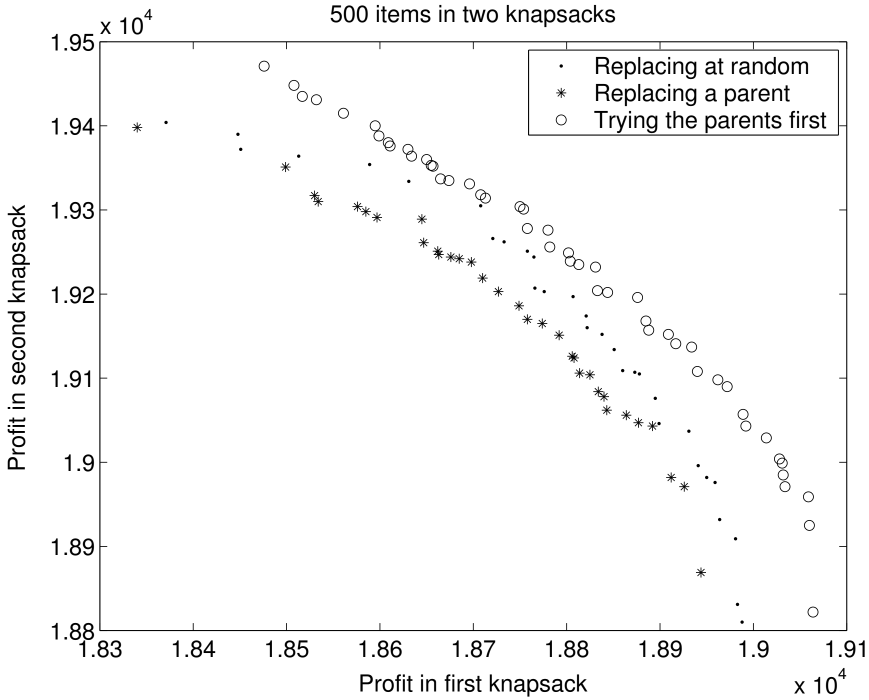
If offspring dominates selected individual

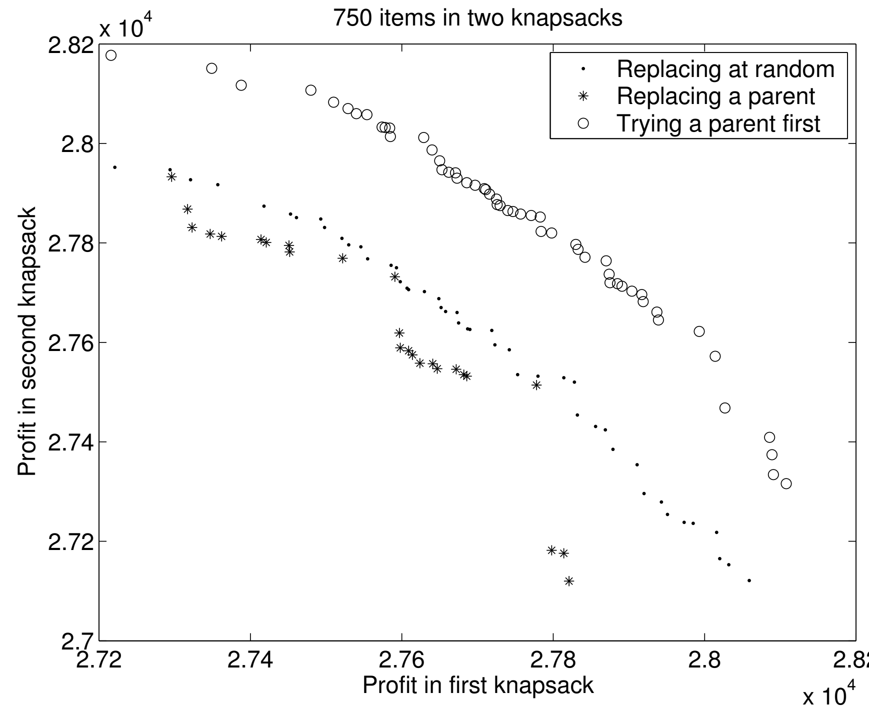
Then offspring replaces it in the population; ****quitloop****

Until all members of population are tried

{offspring dies if it does not replace any member of the population}

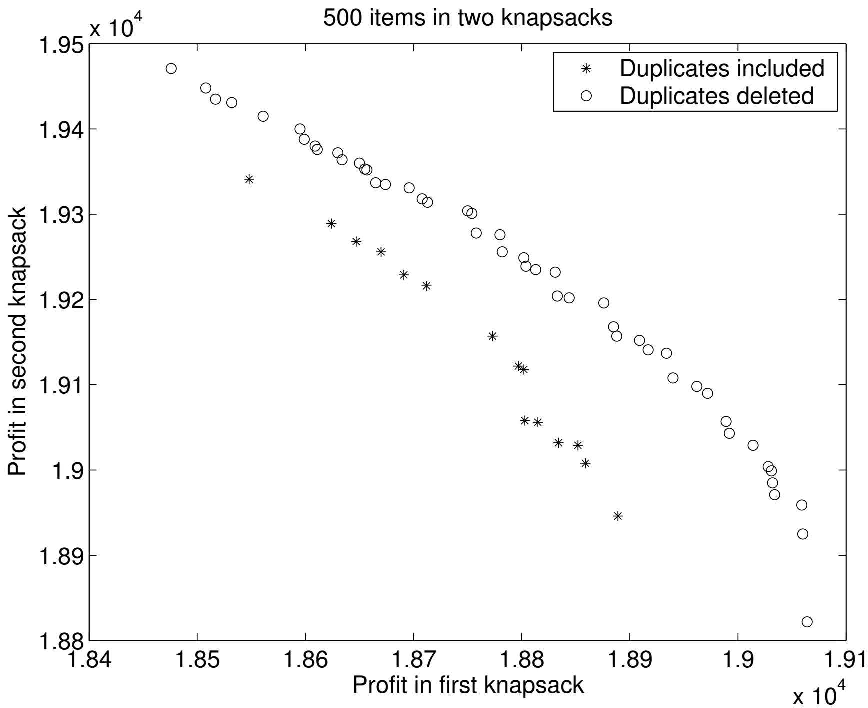
Results for the Simple Strategies

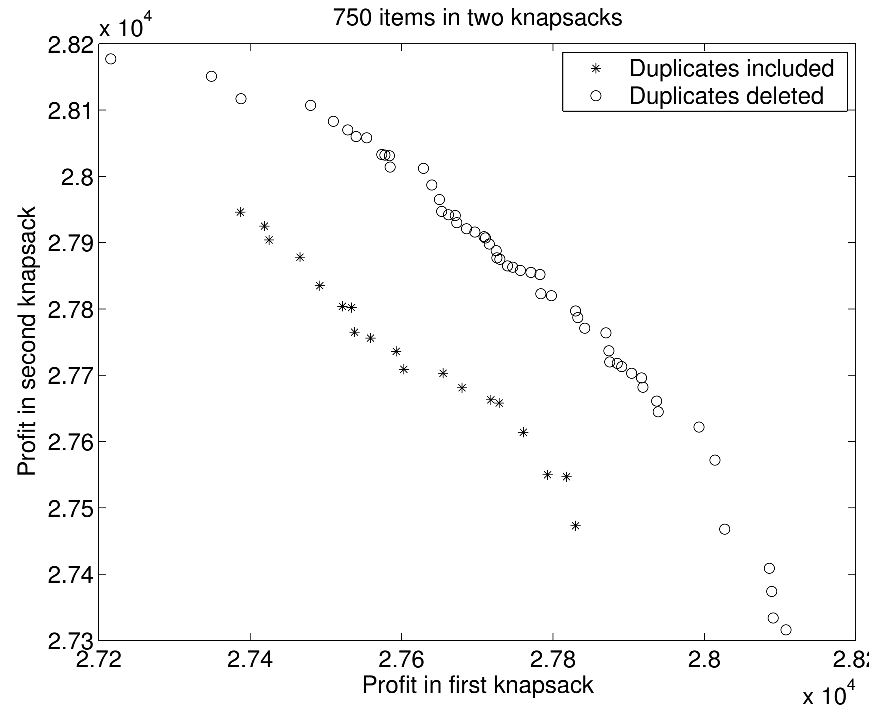




Comparing replacement strategies with duplicates deleted

Results for the Simple Strategies





Examining the effect the deleting duplicates has on the results produced by strategy 3

Results for the Simple Strategies

Average run times of experiments in seconds

Problem	1a	1b	2a	2b	3a	3b
kn500.2	19	19	9	9	19	19
kn750.2	31	32	15	15	31	32

a: duplicates allowed
b: duplicates deleted

Further Strategies

- Strategy 3 is the best simple strategy

Further Strategies

- Strategy 3 is the best simple strategy
- Replacing parent when offspring dominates, to preserve genetic diversity

Further Strategies

- Strategy 3 is the best simple strategy
- Replacing parent when offspring dominates, to preserve genetic diversity
- Otherwise replacing random population member that it dominates

Further Strategies

- Strategy 3 is the best simple strategy
- Replacing parent when offspring dominates, to preserve genetic diversity
- Otherwise replacing random population member that it dominates
- Does it make sense to preserve offspring dominated by both parents?

Strategy 4

- Strategy 4 allows offspring that neither dominate nor are dominated by their parents to live

Strategy 4

- Strategy 4 allows offspring that neither dominate nor are dominated by their parents to live
- But allows offspring that are dominated by both their parents to die

Strategy 4 (cont)

1. **if** offspring dominates either parent, it replaces it

Strategy 4 (cont)

1. **if** offspring dominates either parent, it replaces it
2. **else if** offspring is neither dominated by nor dominates either parent it replaces another individual that it dominates at random

Strategy 4 (cont)

1. **if** offspring dominates either parent, it replaces it
2. **else if** offspring is neither dominated by nor dominates either parent it replaces another individual that it dominates at random
3. **otherwise** it dies

Replacement Strategy 5

1. **if** offspring harbors a new best-so-far Pareto component
    (a) it replaces a parent, if possible
    (b) **else** it replaces another individual at random

Replacement Strategy 5

1. **if** offspring harbors a new best-so-far Pareto component
   (a) it replaces a parent, if possible
   (b) **else** it replaces another individual at random
2. **else if** offspring dominates either parent it replaces it

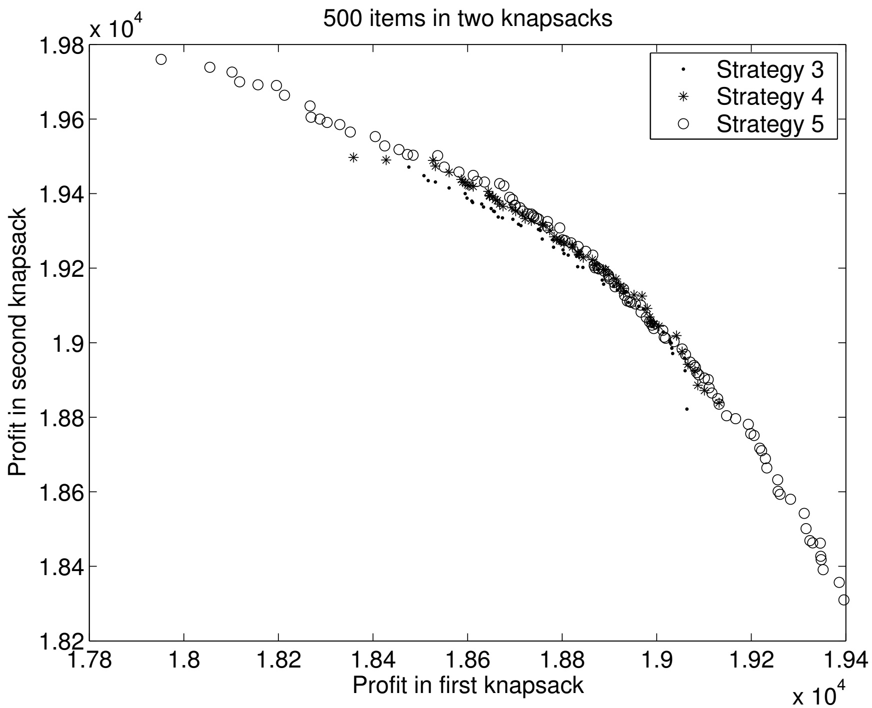
Replacement Strategy 5

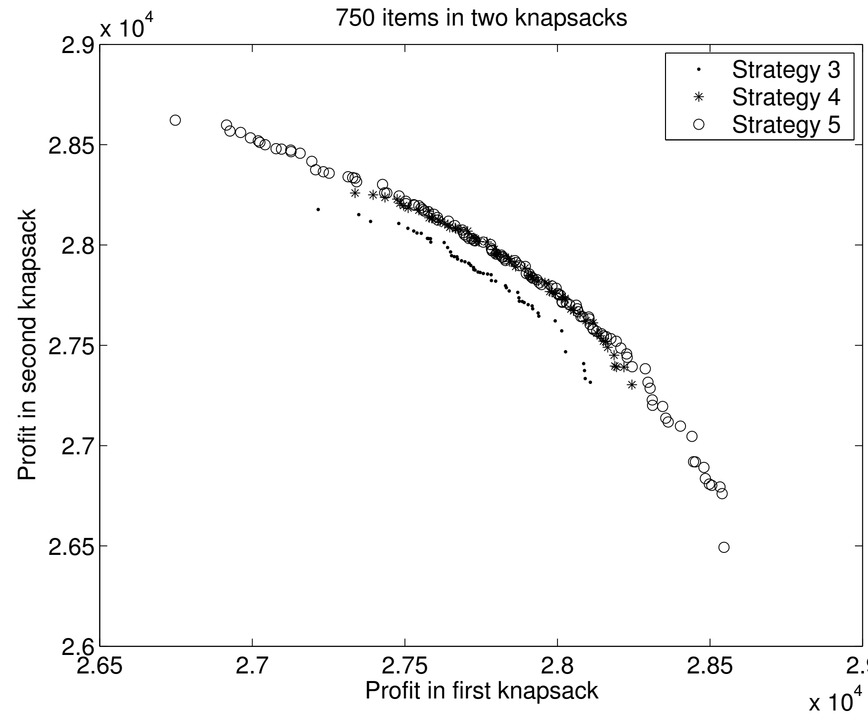
1. **if** offspring harbors a new best-so-far Pareto component
   (a) it replaces a parent, if possible
   (b) **else** it replaces another individual at random
2. **else if** offspring dominates either parent it replaces it
3. **else if** offspring is neither dominated by nor dominates either parent it replaces another individual that it dominates at random

Replacement Strategy 5

1. **if** offspring harbors a new best-so-far Pareto component
   (a) it replaces a parent, if possible
   (b) **else** it replaces another individual at random
2. **else if** offspring dominates either parent it replaces it
3. **else if** offspring is neither dominated by nor dominates either parent it replaces another individual that it dominates at random
4. **otherwise** it dies

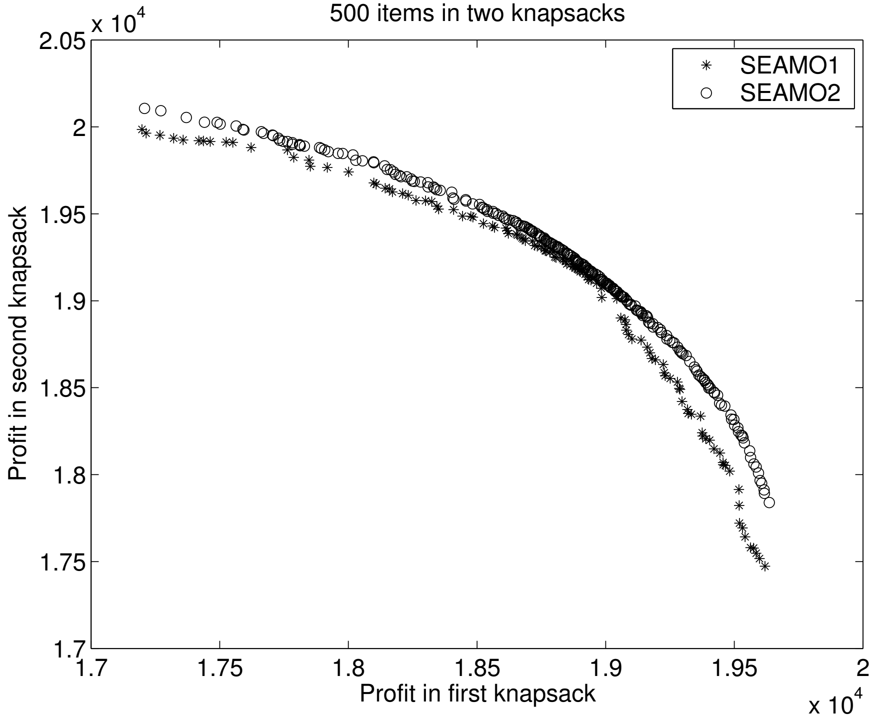
Results for Strategies 4 and 5

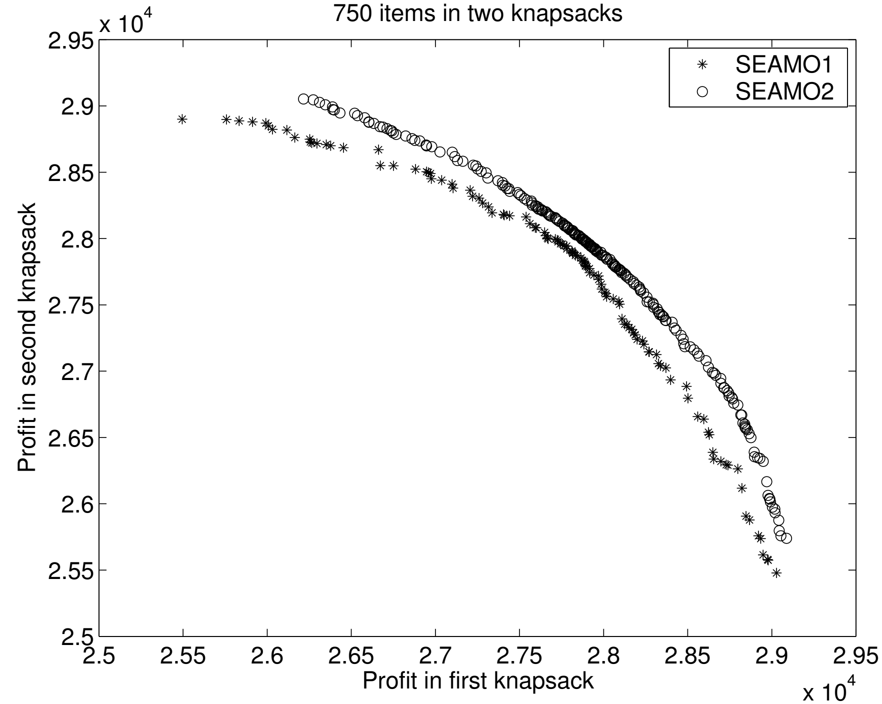




Comparing strategies 3, 4 and 5

Results for Strategies 4 and 5





Comparing SEAMO with strategy 5 (SEAMO2) with the original SEAMO (SEAMO1)

Comparing SEAMO2 with Other EAs

- Compared with NSGA2, PESA and SPEA2 (results downloaded from E. Zitzler's web page)

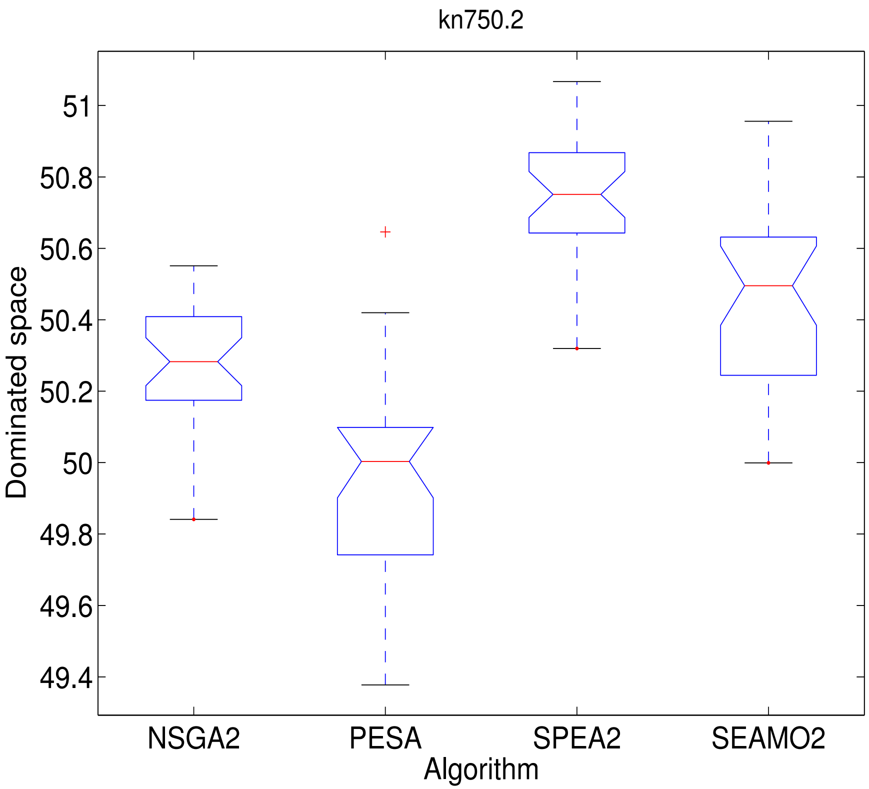
Comparing SEAMO2 with Other EAs

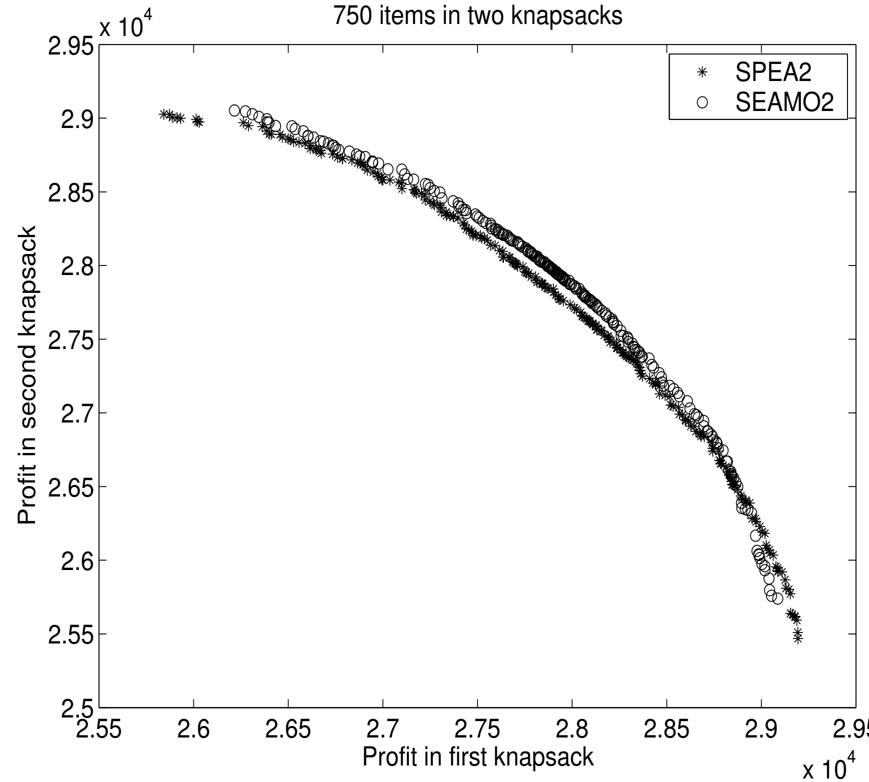
- Compared with NSGA2, PESA and SPEA2 (results downloaded from E. Zitzler's web page)
- For MKP and continuous problems (SPH-2, ZDT6, QV and KUR)

Comparing SEAMO2 with Other EAs

- Compared with NSGA2, PESA and SPEA2 (results downloaded from E. Zitzler's web page)
- For MKP and continuous problems (SPH-2, ZDT6, QV and KUR)
- Population sizes and number of evaluations consistent for EAs

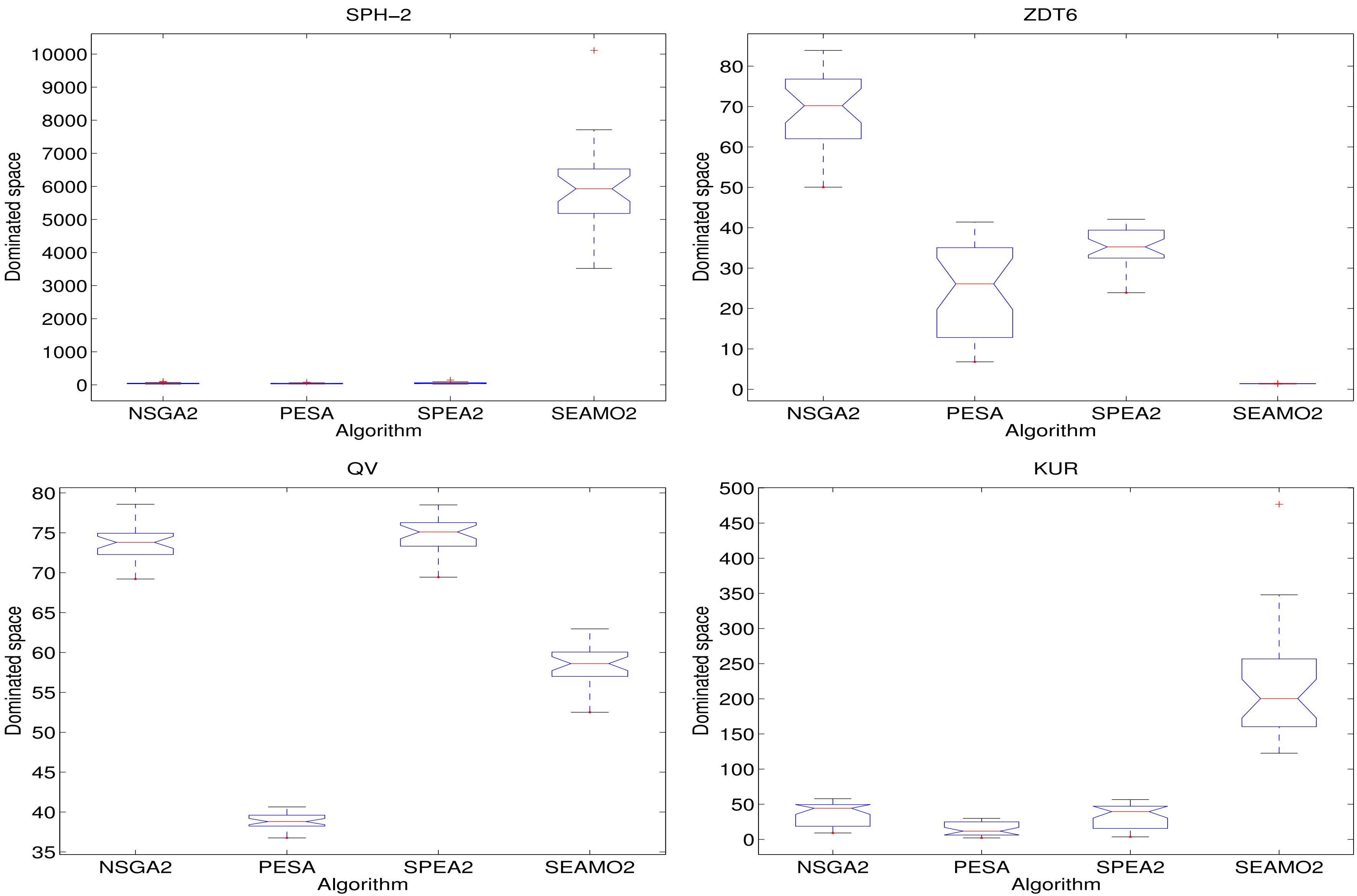
Knapsack Problem, kn750.2





Comparing SEAMO2 with SPEA2

Continuous Problems



Coverage, Coverage $(A \succeq B)$

Average values (and standard deviations) for Coverage $(A \succeq B)$

Coverage $(A \succeq B)$						
Algorithm		Test problems				
A	B	kn750.2	SPH-2	ZDT6	QV	KUR
SEAMO2	NSGA2	73.5 (20.0)	85.5 (14.1	0 (0)	36.9 (11.8)	93.1 (8.9)
	PESA	69.4 (19.4)	88.0 (9.5)	0 (0)	52.1 (11.5)	89.6 (16.8)
	SPEA2	72.5 (13.1)	81.4 (13.4)	0 (0)	35.0 (11.7)	93.4 (7.4)
NSGA2	SEAMO2	11.7 (15.5)	0 (0)	97.7 (0.3)	35.5 (15.7)	0.2 (0.8)
PESA		10.8 (11.8)	0 (0)	96.9 (1.4)	0.23 (0.6)	0.15 (0.8)
SPEA2		9.7 (9.4)	0 (0)	97.7 (0.3)	33.6 (19.7)	0(0)

Summary of Results

- SEAMO2 outperforms its competitors on SPH-2 and KUR for both metrics: dominated space, $\mathcal{S}$, and coverage, Coverage $(A \succeq B)$

Summary of Results

- SEAMO2 outperforms its competitors on SPH-2 and KUR for both metrics: dominated space, $\mathcal{S}$, and coverage, Coverage $(A \succeq B)$
- Additionally, SEAMO2 outperforms the other EAs for coverage,Coverage $(A \succeq B)$, on kn750.2 and QV

Summary of Results

- SEAMO2 outperforms its competitors on SPH-2 and KUR for both metrics: dominated space, $\mathcal{S}$, and coverage, Coverage $(A \succeq B)$
- Additionally, SEAMO2 outperforms the other EAs for coverage,Coverage $(A \succeq B)$, on kn750.2 and QV
- SEAMO2 performs very poorly on ZDT6

Summary of Results

- SEAMO2 outperforms its competitors on SPH-2 and KUR for both metrics: dominated space, $\mathcal{S}$, and coverage, Coverage $(A \succeq B)$
- Additionally, SEAMO2 outperforms the other EAs for coverage,Coverage $(A \succeq B)$, on kn750.2 and QV
- SEAMO2 performs very poorly on ZDT6
- Some caution is required, however, due to some differences in representation and operators

Conclusions

- Some simple evolutionary strategies have been explored for an elitist, steady-state, Pareto-based multi-objective EA

Conclusions

- Some simple evolutionary strategies have been explored for an elitist, steady-state, Pareto-based multi-objective EA
- Leading to an improved version of SEAMO (SEAMO2)

Conclusions

- Some simple evolutionary strategies have been explored for an elitist, steady-state, Pareto-based multi-objective EA
- Leading to an improved version of SEAMO (SEAMO2)
- Despite its simplicity, SEAMO2 is competitive with other state-of-the-art multi-objective EAs

Future Plans

- Development of hierarchical and parallel versions

Future Plans

- Development of hierarchical and parallel versions
- Improving the performance of SEAMO on non-uniformly spread functions such as ZDT6.

Future Plans

- Development of hierarchical and parallel versions
- Improving the performance of SEAMO on non-uniformly spread functions such as ZDT6.
- Applying SEAMO to real world problems