A Simple Evolutionary Algorithm for Multi-Objective Optimization (SEAMO)

Christine L. Valenzuela
Cardiff University, Wales, U.K.
`C.L.Valenzuela@cs.cf.ac.uk`

December 26, 2010

Summary of Paper

- SEAMO is a simple steady-state, Pareto-based evolutionary algorithm
- Uses an elitist strategy for replacement
- Has a simple uniform scheme for selection
- Performs no fitness calculations
- Progress depends entirely on the replacement policy
- Obtained good results for some multiple knapsack problems

Multi-Objective Optimization

- This involves the simultaneous optimization of several objectives
- Characterized by a set of alternative solutions, the *Pareto-optimal set*
- These are *Non-dominated solutions*
  - it is not possible to improve the value of any one of the objectives, in such a solution, without simultaneously degrading the quality of one or more of the other objectives.

Extracting non-dominated solutions with 2 objectives

A maximimzation example:

(2, 10) (9, 5) (4, 5) (5, 7) (10, 4) (1, 9)

Sort on first objective:

(10, 4) (9, 5) (5, 7) (4, 5) (2, 10) (1, 9)

Objective 1 decreasing, objective 2 increasing, leaves:

(10,4) (9, 5) (5, 7) (2, 10)

EA Replacement Rules

1. Parents can be replaced only by their own offspring,
2. Offspring can only replace parents if the offspring are superior – thus the scheme is elitist,
3. Duplicates in the population are deleted.

- rules 1 and 3 to maintain diversity and prevent premature convergence,
- rule 2 to ensure that the best solutions are not lost.

THE 0-1 MULTIPLE KNAPSACK PROBLEM

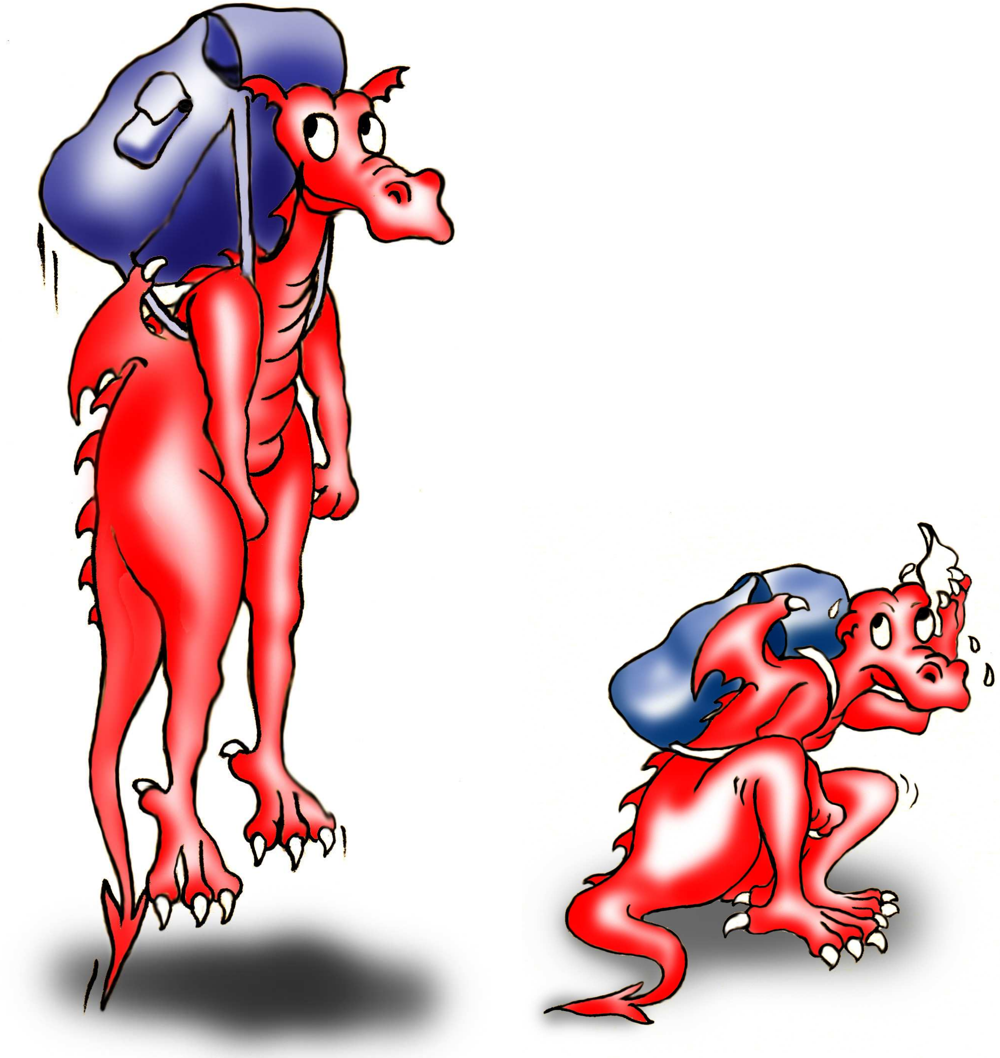
– a generalization of the 0-1 simple knapsack problem:

- A set of objects $O = \{o_1, o_2, o_3, ..., o_n\}$
- And a knapsack of capacity C are given.
- Each object o_i has an associated profit p_i and weight w_i.
- The objective is to find a subset $S \subseteq O$ such that the weight sum over the objects in S does not exceed the knapsack capacity
- And yields a maximum profit.

THE 0-1 MULTIPLE KNAPSACK PROBLEM (MKP)

- The 0-1 MKP involves m knapsacks of capacities $c_1, c_2, c_3, ..., c_m$.
- Every selected object must be placed in all m knapsacks,
- Although neither the weight of an object o_i nor its profit is fixed,
- And will probably have different values in each knapsack.
- The present study is confined to problems involving two knapsacks, i.e. $m = 2$.

The same objects may have different weights in each knapsack



The same objects may have different profits in each knapsack

The Representation and Decoder

- Solutions are represented as simple permutations of the objects to be packed.
- A decoder then packs the individual objects, one at a time, starting at the beginning of the permutation list, and working through.
- For each object that is packed, the decoder checks to make sure that none of the weight limits is exceeded for any knapsack.
- Packing is discontinued as soon as a weight limit is exceeded for a knapsack,
- And when this is detected the final object that was packed is removed from all the knapsacks.

Crossover and Mutation

- Cycle crossover
- The mutation operator swaps two arbitrarily selected objects within a single permutation list.

Procedure *SEAMO*

begin

Generate N random permutations $\{N$ is the population size$\}$

Evaluate the objective vector for each structure and store it

Record the *best-so-far* for each objective function

Repeat

For each member of the population

This individual becomes the first parent

Select a second parent at random

Apply crossover to produce offspring

Apply a single mutation to the offspring

Evaluate the objective vector produced by offspring

If offspring’s objective vector improves on any *best-so-far*

Then it replaces one of the parents and *best-so-far* is updated

Else If offspring dominates one of the parents

Then it replaces it (unless it is a duplicate, then it is deleted)

Endfor

Until stopping condition satisfied

Print all non-dominated solutions in the final population

End

The Test Problems

- The four test problems of Zitzler and Thiele with two knapsacks are used here.
- With 100, 250, 500 or 750 objects.
- Restricting the test-bed to two knapsacks means that solutions can be plotted using standard 2D graphics, and their quality easily visualized.

Elitist Strategy

- The idea is to breed a diverse population of solution pairs that is as close to the Pareto front as is possible.
- The dual aims pursued during the search process are:
  1. To move the current solutions in the population ever closer to the Pareto front, and
  2. To extend the diversity of the solution set by improving on the individual global best profits for knapsack 1 and knapsack 2.
- Improvements in both (1) and (2) are achieved by the replacement strategy used in SEAMO, and not by the selection process.

Selection Procedure for SEAMO

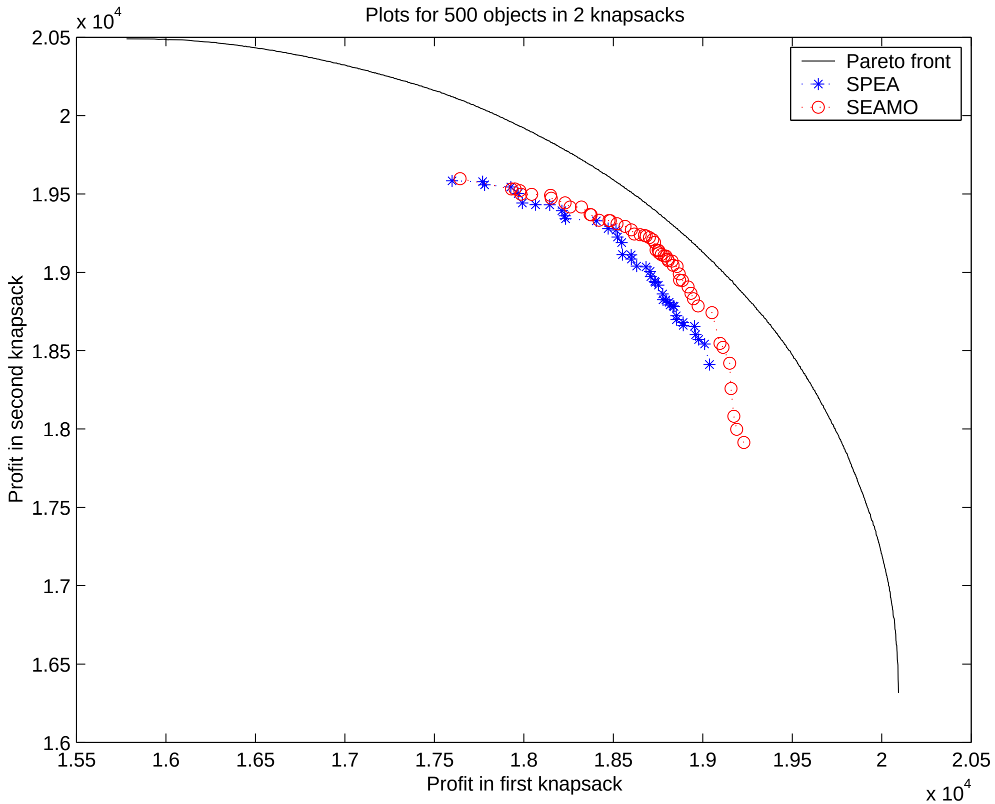
- The selection procedure for SEAMO does not rely on fitness calculations or dominance relationships.
- Each individual in the population serves as the first parent once,
- And the second parent is then selected at random (uniformly).
- Objective values and dominance relationships are only considered at the replacement stage,
- If an offspring dominates one of its parents or includes a new global best for one of the knapsacks, it replaces its parent.

Results

- SEAMO is compared with Zitler and Thiele's SPEA algorithm
- Previously published results showed SPEA better than other EAs on knapsack problems: VEGA, HLGA, NPGA, NSGA

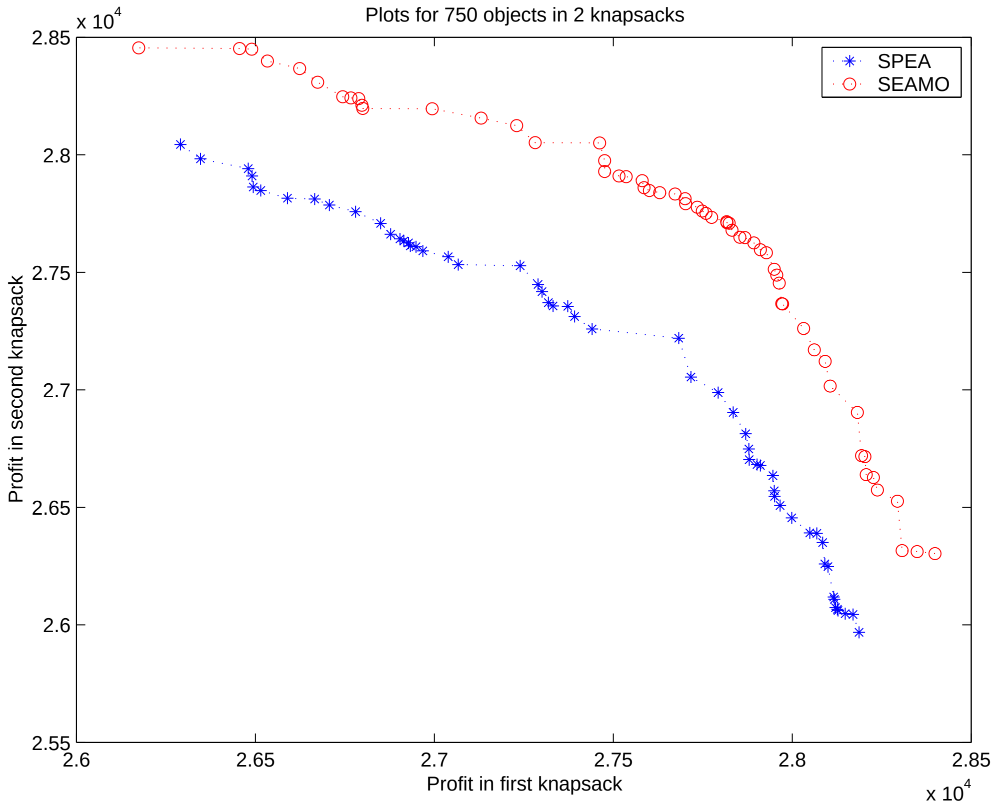
500 objects in 2 knapsacks

Figure 2:Non-dominated solutions from 30 replicate runs of SEAMO and SPEA.



500 objects in 2 knapsacks

Figure 3:Non-dominated solutions from 30 replicate runs of SEAMO and SPEA.



Summary of Other Results in Paper

Can improve results further by:

- Increasing population size
- Running EA for longer time

Further Work on SEAMO

- Comparisons with more recent MO EAs – SPEA2, PESA and NSGAII
- Knapsack problems with 3 and 4 objectives (knapsacks)
- Continuous multiobjective functions

Results for knapsack problems with 750 objects

To outperform its competitors SEAMO needs to perform about:

- 2 knapsacks: 4 times as many evaluations
- 3 knapsacks: twice as many evaluations
- 4 knapsacks: the same number of evaluations

SEAMO improves relative to other EAs as number of objectives increases.

The Continuous Test Functions

n Type	Domain	Objective functions
SPH-m (Schaffer 1985; Laumans, Rudolph, and Schwefel 2001)		
100 min	$[-10^3, 10^3]^n$	$f_j(x) = \sum_{1\le i\le n, i\ne j} (x_i)^2 + (x_j - 1)^2$ $1 \le j \le m,\ m = 2$
ZDT6 (Zitzler, Deb and Thiele 2000)		
100 min	$[0, 1]^n$	$f_1(x) = 1 - \exp(-4x_1) sin^6(6\pi x_1)$ $f_2(x) = g(x)[1 - (f_1(x)/g(x))^2]$ $g(x) = 1 + 9.((\sum_{i=2}^{n} x_i/(n-1))^{0.25}$
QV (Quagliarella and Vicini 1997)		
100 min	$[-5, 5]^n$	$f_1(x) = (1/n \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10))^{1/4}$ $f_2(x) = (1/n \sum_{i=1}^{n} ((x_i - 1.5)^2 - 10\cos(2\pi(x_i - 1.5)) + 10))^{1/4}$
KUR (Kursawe 1991)		
100 min	$[-10^3, 10^3]^n$	$f_1(x) = \sum_{i=1}^{n} (\lvert x_i\rvert^{0.8} + 5.\sin^3(x_i) + 3.5828)$ $f_2(x) = \sum_{i=1}^{n-1} (1 - \exp^{-0.2\sqrt{x_i^2 + x_{i+1}^2}})$

Results for Continuous Functions

- SEAMO does better than competitors on 3 out of 4 of functions – SPH-2, QV, and KUR
- SEAMO no use on ZDT6

Future Work

This will concentrate on:

- Improving the performance of SEAMO, without compromising its simplicity
- Extending it to real world applications