European Journal of Operational Research 213 (2011) 538–550

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Interfaces with Other Disciplines

Heuristic algorithms for the cardinality constrained efficient frontier

M. Woodside-Oriakhi, C. Lucas, J.E. Beasley *

CARISMA, Mathematical Sciences, Brunel University, Uxbridge UB8 3PH, United Kingdom

ARTICLE INFO

Article history:
Received 12 February 2010
Accepted 15 March 2011
Available online 22 March 2011

Keywords:
Efficient frontier
Genetic algorithm
Portfolio optimisation
Simulated annealing
Tabu search

ABSTRACT

This paper examines the application of genetic algorithm, tabu search and simulated annealing metaheuristic approaches to finding the cardinality constrained efficient frontier that arises in financial portfolio optimisation. We consider the mean–variance model of Markowitz as extended to include the discrete restrictions of buy-in thresholds and cardinality constraints. Computational results are reported for publicly available data sets drawn from seven major market indices involving up to 1318 assets. Our results are compared with previous results given in the literature illustrating the effectiveness of the proposed metaheuristics in terms of solution quality and computation time.

1. Introduction

As billions of dollars are invested in markets around the world, investors must not only consider maximising their expected return, but also minimising the volatility that results from expected fluctuations in the value of their investment portfolios. Increasingly, portfolio managers are seeking more robust asset selection (portfolio formation) strategies to create desirable portfolios for their investors. More formally we can define a desirable portfolio as one that potentially gives a good tradeoff between investment risk and return.

Markowitz (1952) set up a clear quantitative framework for the selection of a portfolio, summarising the process of portfolio selection as an allocation of resources so as to tradeoff expected return and risk. Through the use of statistical measurements of expectation and variance of return (variance being equated to risk), Markowitz described the benefit and risk associated with an investment.

In order to capture tradeoff (in a single period static portfolio planning situation) two approaches are possible:

- Minimise the risk of the portfolio for a given level of expected return. This entails solving a mathematical optimisation problem with continuous variables, a quadratic objective and linear constraints.
- Maximise the expected level of return for a given level of risk. This entails solving a mathematical optimisation problem with continuous variables, a linear objective and linear constraints but with one quadratic equality constraint.

Although these two approaches are logically equivalent the approach that is more effective computationally is to minimise the risk of the portfolio for a given level of return. This is because problems with quadratic objectives are easier to handle numerically than problems with quadratic constraints (Hillier and Lieberman, 2010). Hence by formulating and solving a parametric quadratic program (QP), Markowitz determines an ***efficient frontier*** as the set of (undominated) portfolios found by minimising risk (variance) as you vary the desired return.

Note here that one view of the problem that can be adopted is that there are two objectives, namely (maximise return, minimise risk), and so multiobjective solution approaches can be applied. A number of such approaches can be found in our literature survey below.

Markowitz's approach has become the core decision engine of many portfolio analytic and planning systems in constructing efficient frontiers, which can be viewed as the set of Pareto optimal (expected return, variance of return) combinations under conditions of uncertainty. The standard Markowitz model assumes a perfect market without transaction costs or taxes where short selling is not permitted, but assets are infinitely divisible and can therefore be traded in any non-negative proportion. The beauty of this simplistic unconstrained risk-return model is that it is capable of being extended to capture market realism. However, the introduction of a single cardinality constraint restricting the number of assets present in the portfolio changes the classical quadratic optimisation model to a quadratic mixed-integer problem (QMIP) that is NP-hard (Moral-Escudero et al., 2006). As QMIPs are hard to solve optimally many practitioners and researchers have used heuristics, i.e. non-exact methods, in this area.

This paper emphasises finding the cardinality constrained efficient frontier (CCEF) using metaheuristic approaches, namely

* Corresponding author. Tel.: +44 1895 266219; fax: +44 1895 269732.
E-mail addresses: maria.woodsideoriakhi@brunel.ac.uk (M. Woodside-Oriakhi), cormac.lucas@brunel.ac.uk (C. Lucas), john.beasley@brunel.ac.uk (J.E. Beasley).

0377-2217/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2011.03.030

a genetic algorithm, tabu search and simulated annealing. In terms of coding we use a modeling language for mathematical programming (AMPL; Fourer et al., 2002).

The remainder of this paper is organised as follows. In Section 2, we present a formulation of the cardinality constrained portfolio optimisation problem. A literature review of exact and heuristic algorithms for the problem is presented in Section 3. The heuristics of genetic algorithm, tabu search and simulated annealing are introduced in Section 4, together with their algorithmic application to the problem under consideration. Then, in Section 5, we present computational results for data sets taken from seven major stock market indices. We provide conclusions in Section 6.

2. Formulation

Let:

N be the total number of assets available,
μ_i be the expected return of asset i $(i = 1,\ldots,N)$,
σ_{ij} be the covariance between the return of asset i and asset j $(i = 1,\ldots,N, j = 1,\ldots,N)$,
ρ be the desired level of expected return,
K be the desired number of assets in the chosen portfolio,
l_i $(\geqslant 0)$ be the minimum proportion of the total investment held in asset i $(i = 1,\ldots,N)$, if any investment is made in asset i, and
u_i $(\geqslant 0)$ be the maximum proportion of the total investment that can be held in asset i, $(i = 1,\ldots,N)$.

The decision variables are:

x_i the proportion $(0 \leqslant x_i \leqslant 1)$ of the total investment held in asset i $(i = 1,\ldots,N)$, and
δ_i which is 1 if any of asset i $(i = 1,\ldots,N)$ is held, 0 otherwise.

The cardinality constrained portfolio optimisation problem is

$$\text{Minimise} \quad \sum_{i=1}^{N}\sum_{j=1}^{N}\sigma_{ij}x_i x_j, \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{N}\mu_i x_i = \rho, \tag{2}$$

$$\sum_{i=1}^{N} x_i = 1, \tag{3}$$

$$l_i\delta_i \leqslant x_i \leqslant u_i\delta_i, \quad i = 1,\ldots,N, \tag{4}$$

$$\sum_{i=1}^{N}\delta_i = K, \tag{5}$$

$$\delta_i = 0 \quad \text{or} \quad 1, \quad i = 1,\ldots,N. \tag{6}$$

Eq. (1) involves the covariance matrix to minimise the volatility (variance) associated with the chosen portfolio. Eq. (2) ensures that the portfolio has an expected return of ρ, whilst Eq. (3) ensures that the investment proportions sum to one. Eq. (4) is the buy-in threshold restricting asset investments. In this equation if an asset i is not held, $\delta_i = 0$, then the resulting proportion x_i is also zero. If an asset i is held, $\delta_i = 1$, then the equation ensures that the investment proportion lies between the appropriate lower and upper limits, $l_i \leqslant x_i \leqslant u_i$. Eq. (5) is the cardinality constraint ensuring that there are exactly K assets in the portfolio. Eq. (6) is the integrality constraint, reflecting the inclusion or exclusion of an asset.

Note here that although we have formulated the problem above using covariances an equivalent formulation can be obtained using correlations. This arises since the covariance between the returns of assets i and j is equal to the product of the standard deviations in return for assets i and j multiplied by the correlation between returns for assets i and j.

This optimisation model, Eqs. (1)–(6), is a quadratic mixed-integer program that has been given previously in Chang et al. (2000). It is appropriate to use exactly the same formulation as they used since in this paper we intend to make a direct computational comparison between our work and the work of Chang et al. (2000). The Markowitz unconstrained model is simply Eqs. (1)–(4) with $\delta_i = 1$ $i = 1,\ldots,N$.

A variant of the problem given in Eqs. (1)–(6) that is encountered in the literature is where the equality constraint seen in Eq. (5) is relaxed to an inequality, so $\sum_{i=1}^{N}\delta_i \leqslant K$. Our focus in this paper however is on the problem as defined by Chang et al. (2000) where we seek precisely K assets in the portfolio.

As for the Markowitz unconstrained model it is possible to generate an efficient frontier by minimising risk (Eq. (1)) for varying values of the desired expected return ρ. However it is now well known that through the introduction of discrete constraints, Eqs. (4)–(6), discontinuities are seen in an otherwise continuous efficient frontier.

Table 1 gives the data for the $N = 4$ asset example of Chang et al. (2000). For this data Fig. 1 shows the unconstrained efficient frontier (UEF) and the cardinality constrained efficient frontier (CCEF) where $K = 2$. Note that although the UEF is continuous, the CCEF is not. For more as to the discontinuous nature of the CCEF see Chang et al. (2000).

The above QMIP problem (Eqs. (1)–(6)) for cardinality constrained portfolio optimisation can potentially be solved by modern optimisation packages such as CPLEX. However, in practice, as results presented in Section 5 will indicate, this is not a computationally effective approach as problem size increases. In the light of this the majority of work that has been presented in the literature has focused on heuristics for the problem. In the next section we review the work that has been presented in the literature relating to cardinality constrained portfolio optimisation.

Table 1
Data for UEF and CCEF.

Asset	Return	Standard deviation	Correlation matrix			
			1	2	3	4
1	0.004798	0.046351	1	0.118368	0.143822	0.252213
2	0.000659	0.030586		1	0.164589	0.099763
3	0.003174	0.030474			1	0.083122
4	0.001377	0.035770				1

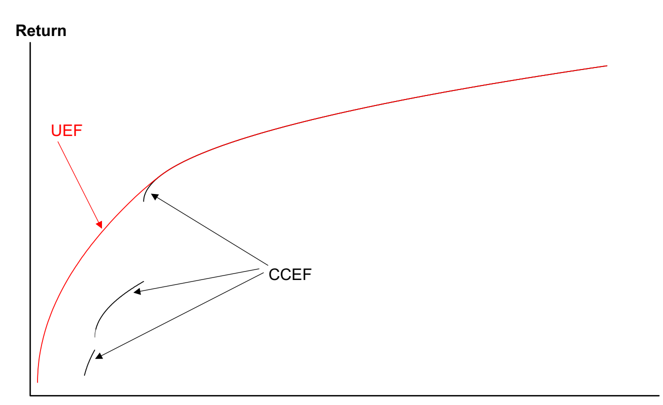

Fig. 1. The UEF and CCEF for a four asset example.

3. Literature survey

In order to structure our literature survey we consider exact and heuristic approaches for cardinality constrained portfolio optimisation separately.

We would comment here that although we are aware that there are papers in the literature that deal with constrained portfolio optimisation (e.g. Bonami and Lejeune, 2009; Duran et al., 2009; Corazza and Favaretto, 2007; Kellerer et al., 2000; Mansini and Speranza, 1999; Syam, 1998) we, for reasons of space, only review below papers that include a cardinality constraint relating to the number of assets in the portfolio.

3.1. Exact approaches

Bienstock (1996) presents a branch and cut algorithm for the exact solution of the cardinality constrained portfolio optimisation problem. The cardinality constraint (Eq. (5)) is an inequality rather than an equality. Computational results are given for some real-life data sets.

Li et al. (2006) present an approach for the exact solution of the cardinality constrained portfolio optimisation problem when the amounts to be invested in each asset must be in specified lots. Any money not invested in assets is invested at a risk free rate. In their approach the cardinality constraint (Eq. (5)) is an inequality rather than an equality. Their approach is a convergent Lagrangian and contour-domain cut method. Computational results are given for one problem involving 30 assets taken from the Hong Kong market.

Shaw et al. (2008) present a lagrangean relaxation based procedure for the exact solution of the cardinality constrained portfolio optimisation problem. The cardinality constraint (Eq. (5)) is an inequality rather than an equality. In their approach the covariance matrix is decomposed into a diagonal asset risk matrix and a covariance matrix for the F factors adopted. This reduces the size of the quadratic term in the objective from N^2 to F^2. A well-known US equity model has $F = 68$ for example. Computational results are reported for eight test problems involving up to 500 assets. They report that CPLEX (version 8.1) failed to solve any of these problems to proven optimality in four hours of computation. By contrast their approach solved seven of the eight test problems.

Vielma et al. (2008) present a branch-and-bound algorithm for the exact solution of the cardinality constrained portfolio optimisation problem based on a lifted polyhedral relaxation of conic quadratic constraints. The cardinality constraint (Eq. (5)) is an inequality rather than an equality. Computational results are presented for problems drawn from real-world data.

Bertsimas and Shioda (2009) present an approach for the exact solution of the cardinality constrained portfolio optimisation problem. In their approach the cardinality constraint (Eq. (5)) is an inequality rather than an equality. They use Lemkes pivoting algorithm (Lemke and Howson, 1964) to solve successive subproblems in the search tree. Computational results are presented for their approach as well as for CPLEX on problems involving up to 500 assets. One feature of their results is that for all of the portfolio optimisation test problems considered both their approach and CPLEX (version 8.1) failed to find even a single provably optimal solution within the computational time limit they allow (either two minutes or one hour depending on the size of the problem).

Gulpinar et al. (2010) present an approach for the exact solution of the cardinality constrained portfolio optimisation problem. In their approach, based on the difference of convex functions programming, the cardinality constraint (Eq. (5)) is an equality. They select a portfolio with regard to the worst-case associated with specified scenarios. Computational results are given for selecting portfolios of varying cardinality from a universe of 98 assets.

3.2. Heuristic approaches

Chang et al. (2000) illustrate the discontinuous nature of the efficient frontier in the presence of cardinality restrictions and present three heuristic algorithms based upon a genetic algorithm, tabu search and simulated annealing for finding the cardinality constrained efficient frontier. Computational results are presented for five test problems (that are made publicly available) involving up to 225 assets.

Following the work of Chang et al. (2000) papers relating to heuristic approaches can be subdivided into two, those that apply just a single metaheuristic, and those that apply two or more metaheuristics (as in Chang et al., 2000). We structure our review below accordingly.

3.2.1. Single metaheuristic approaches

Crama and Schyns (2003) present a simulated annealing approach. As well as a cardinality constraint they include constraints on turnover and trading related to the presence of an existing portfolio. Constraint violations are dealt with using a penalty function related to the magnitude of the violation raised to a power. Computational results are given for one test problem involving 151 assets.

Derigs and Nickel (2003) present a simulated annealing based metaheuristic. In their approach stock returns and covariances are derived from a linear multi-factor model, where the factors are based on macro-economic variables. They present a case study based around an investment trust tracking the German DAX30 index. Their investment universe, some 202 stocks, was taken from the DAX30 and STOXX200. Limited computational results are presented. More as to their work can be found in Derigs and Nickel (2004).

Moral-Escudero et al. (2006) present a genetic algorithm for the problem that uses two different crossover operators (random respectful recombination and random assorting recombination). Computational results are presented that make use of the test problems provided by Chang et al. (2000).

Streichert and Tanaka-Yamawaki (2006) combine a multiobjective evolutionary algorithm with QP local search. In their algorithm a variety of portfolios, each containing K assets, are generated. The proportion invested in each of the K assets is decided by solving a QP. Computational results are given for two of the five test problems used in Chang et al. (2000) involving up to 85 assets.

Fernandez and Gomez (2007) apply a Hopfield neural network to the problem. They also implement (albeit with minor changes) the three heuristics given in Chang et al. (2000). Computational results are presented that make use of the test problems provided by Chang et al. (2000) which indicate that no one heuristic outperforms the others.

Chiam et al. (2008) present an approach based upon a multiobjective evolutionary algorithm. Computational results are presented that make use of the test problems provided by Chang et al. (2000).

Branke et al. (2009) use a multiobjective evolutionary algorithm in conjunction with the critical line algorithm of Markowitz (1956). They include a constraint (involving additional zero-one variables) based on German investment law. Computational results are given for three of the five test problems from Chang et al. (2000), as well as for one further problem involving 500 assets.

Chang et al. (2009) present a genetic algorithm for the problem. In their model they replace the objective by a weighted sum of risk and return. They also consider measures of risk other than variance (e.g. semi-variance, mean absolute deviation, skewness). They report results for three test problems involving up to 99 assets.

Pai and Michel (2009) apply a clustering approach to choosing the assets to include in the portfolio, thereby eliminating the cardinality constraint. They use an evolutionary strategy approach to decide the proportion to be invested in each of the assets. Computational results are presented for data drawn from the Bombay and Tokyo stock markets involving up to 225 assets.

Soleimani et al. (2009) present a genetic algorithm for the problem. Their model includes constraints on the proportion invested in sectors (sets of assets). They present computational results for a number of test problems involving up to 2000 assets.

Anagnostopoulos and Mamanis (2010) adopt a tri-objective view of the problem and apply three multiobjective evolutionary optimisation algorithms, specifically the non-dominated sorting genetic algorithm II (NSGA-II), the strength pareto evolutionary algorithm 2 (SPEA2) and the Pareto Envelope-based Selection Algorithm (PESA). Computational results are presented for two randomly generated problems involving 200 and 300 assets.

3.2.2. Multiple metaheuristic approaches

Jobst et al. (2001) examine a number of algorithmic options (such as integer restart and reoptimisation) within an existing QP solver, FORTMP (Ellison et al., 1999). Computational results are presented that make use of the test problems provided by Chang et al. (2000). The largest problem solved (225 assets) required over 5 h of computation using their integer restart heuristic.

Schaerf (2002) presents hill climbing (local search), tabu search and simulated annealing algorithms for the problem. A variety of moves relating to the proportion invested in each asset are considered. Computational results are presented that make use of the test problems provided by Chang et al. (2000).

Maringer and Kellerer (2003) present an approach based on combining simulated annealing with evolutionary ideas. They maintain a population of portfolios that are improved in a simulated annealing fashion. As is normal in evolutionary approaches poor portfolios in the population are replaced by better portfolios. Computational results are presented for two test problems involving 30 and 96 assets.

Ehrgott et al. (2004) present an approach using multicriteria decision making. In their problem they have a number of additional portfolio objectives (for example relating to dividends paid and Standard and Poors rating) and these are combined via weighted utility functions. They apply four different heuristic solution techniques (local search, simulated annealing, tabu search, genetic algorithm) to four test problems, involving up to 1416 assets.

Cura (2009) presents an approach based on particle swarm optimisation where each particle represents a portfolio. If a portfolio does not contain the appropriate number of assets then assets are added/deleted from the portfolio. Computational results are presented that make use of the test problems provided by Chang et al. (2000). They also report results for the same test problems using a genetic algorithm, tabu search and simulated annealing which indicate that no one heuristic outperforms the others.

Ruiz-Torrubiano and Suarez (2010) present approaches based on preprocessing (pruning), simulated annealing, genetic algorithms and estimation of distribution algorithms (Larranaga and Lozano, 2001). Computational results are presented that make use of the test problems provided by Chang et al. (2000). They conclude that approaches based on estimation of distribution algorithms do not work well when the number of assets is large.

3.3. Comment

It can be seen from the above literature review that the cardinality constrained portfolio optimisation problem has attracted a reasonable amount of attention in the literature, especially since the work of Chang et al. (2000). Given the computational difficulty of tackling the problem exactly many metaheuristics (as discussed above) have been applied to the problem. Our work complements this body of literature, adding to this literature in two respects:

- Within our metaheuristics we solve, to optimality, a (small) mixed-integer quadratic optimisation problem.
- We present better quality results on publicly available test problems than have been presented before in the literature.

We would comment here that incorporating within a metaheuristic an algorithmic step involving the ***optimal*** solution of an integer program is relatively uncommon in the literature. In particular we are not aware of it being employed in the context of cardinality constrained portfolio optimisation before. However we believe that the quality of our results indicates that it can be a useful strategy to employ.

4. Heuristics for the CCEF

In this section we present our heuristic algorithms for finding the cardinality constrained efficient frontier. We first present the optimisation problem (denoted the subset optimisation problem) that underlies each of our heuristics. Then we present our heuristics which are based on genetic algorithms, tabu search and simulated annealing. For each of these heuristics we first give a brief overview of the general approach before giving the particular implementation of the heuristic that we adopted for the problem under consideration, finding a cardinality constrained efficient frontier.

4.1. Subset optimisation

The heuristics we present in this paper make use of subset optimisation. By this we mean that we specify subsets of assets for which we know their status (either in or out of the chosen portfolio). Given these subsets we optimise for any remaining assets to see if they are in/out of the chosen portfolio. For all assets in the portfolio the proportion invested in the asset is decided by optimisation. In addition we relax the constraint upon desired return such that return is no longer specified precisely, rather we allow return to be in a specified range.

Let ρ be the desired return level, as Eq. (2). Early computational experience indicated that attempting to find a portfolio with precisely K assets and precise return ρ was relatively time-consuming, even if the number of assets from which are choosing is small. For this reason we (in the subset optimisation problem below) solve the problem with an inequality for desired return.

Let $[\rho_L, \rho_U]$ be the return range, so we are content with a portfolio whose return lies in this range. Let S_{in} be the subset of assets that must be included in the chosen portfolio, and S_{out} be the subset of assets that must be excluded from the chosen portfolio, where $S_{\text{in}} \cap S_{\text{out}} = \emptyset$.

Then the subset optimisation problem that we solve is:

$$\text{Minimise} \quad \sum_{i=1}^{N}\sum_{j=1}^{N}\sigma_{ij}x_i x_j, \tag{7}$$

$$\text{subject to} \quad \rho_L \leqslant \sum_{i=1}^{N}\mu_i x_i \leqslant \rho_U, \tag{8}$$

$$\sum_{i=1}^{N}x_i = 1, \tag{9}$$

$$l_i\delta_i \leqslant x_i \leqslant u_i\delta_i, \quad i = 1,\ldots,N, \tag{10}$$

$$\sum_{i=1}^{N} \delta_i = K, \tag{11}$$

$$\sum_{i \in S_{\text{in}}} \delta_i = \min[|S_{\text{in}}|, K], \tag{12}$$

$$\delta_i = 0 \quad \forall i \in S_{\text{out}}, \tag{13}$$

$$\delta_i = 0 \quad \text{or} \quad 1, \quad i = 1, \ldots, N, \tag{14}$$

Eqs. (7) and (9), (10), (11) are as Eqs. (1) and (3), (4), (5). Eq. (8) constrains the expected return to be within the desired range and equations (12) and (13) ensure that assets are set in/out of the portfolio as desired. Eq. (12) forces all assets in S_{in} into the portfolio if $|S_{\text{in}}| \leqslant K$, and chooses K assets from S_{in} if $|S_{in}| > K$. This problem, Eqs. (7)–(14) is also a QMIP, but provided that the number of assets for which we have to make a decision as to whether they are in or out of the portfolio is small (i.e. $N - |S_{in} \cup S_{\text{out}}|$ is small) it can be solved relatively quickly to proven optimality.

For simplicity of notation in the heuristics we present later below we refer to the above subset optimisation problem as $F(S_{\text{in}}, S_{\text{out}})$. An advantage of our approach is that by using a return range $[\rho_L, \rho_U]$ we can ensure that any frontier found covers the return range. This contrasts with other approaches, e.g. Chang et al. (2000), where there is no direct control over the return range covered. In the computational results reported later below we use both $[\rho_L = 0.9\rho, \rho_U = 1.1\rho]$, so a portfolio within ten percent of the desired return, and $[\rho_L = -\infty, \rho_U = +\infty]$, so disregarding desired return.

Note here that one of the potential practical advantages of our heuristics is that any additional (user specified) constraints on the composition of the chosen portfolio can be included in the subset optimisation problem. Such constraints might include, for example:

- Class/sector constraints which specify minimum/maximum exposure to certain sectors (sets of assets).
- Lot size constraints which specify that the amount invested in any asset must be an integer multiplier of a known constant.
- Fixed costs associated with the inclusion of an asset in the portfolio.

The heuristics outlined below are applicable, without significant change, to problems of these types.

4.2. Genetic algorithms

Genetic algorithms (GAs) are a search mechanism based on the evolutionary principles of natural selection and genetics. The theoretical foundations of GAs were originally developed by Holland (1975). They work with a population of solutions and employ the principle of survival of the fitness.

In a GA the decision variables are encoded into finite strings referred to as *chromosomes*. To implement natural selection and evolve good solutions, the chromosomes are evaluated by a fitness criteria. In optimisation problems, such as we consider here, the fitness measure is typically directly related to the objective function (possibly penalised by constraint violation).

GAs rely on a candidate population (typically of fixed size), which they maintain throughout. GAs use four main operators of selection, crossover, mutation and replacement. The population changes through repetition of these operators, with stronger fitter solutions (population members) replacing weaker ones.

For a more comprehensive overview of GAs see Burke and Kendall (2005), Aarts and Lenstra (2003), Beasley (2002), Mitchell (1996).

In the literature, as surveyed above, examples of the application of GAs (evolutionary approaches) to the cardinality constrained portfolio optimisation problem can be found in Chang et al. (2000), Maringer and Kellerer (2003), Ehrgott et al. (2004), Moral-Escudero et al. (2006), Streichert and Tanaka-Yamawaki (2006), Chiam et al. (2008), Branke et al. (2009), Chang et al. (2009), Cura (2009), Pai and Michel (2009), Soleimani et al. (2009), Anagnostopoulos and Mamanis (2010), Ruiz-Torrubiano and Suarez (2010).

4.3. A genetic algorithm for the CCEF

In our GA we use a fixed population size of $P = 100$ portfolios. Given the desired return of ρ each member of the initial population is generated by randomly choosing $max[2K, 20]$ assets to be in S_{in}, all other assets being in S_{out} and then solving the subset optimisation problem $F(S_{\text{in}}, S_{\text{out}})$. In order to try and ensure that the subset optimisation problem is feasible in making a random choice of assets we include in S_{in} some assets i that have return $\mu_i \geqslant \rho$ and some assets i that have return $\mu_i \leqslant \rho$.

In our GA we use parent sets. We first create two parents sets Q_1 and Q_2 (each of fixed size q, in our results below we use $q = 5$). We create the parent sets by sorting the members of the population into increasing risk (variance) order. Take the first $2q$ portfolios in this ordered list and assign the first portfolio to Q_1, the second to Q_2, the third to Q_1, etc. in an alternate fashion. These two sets collectively contain the $2q$ fittest members of the population (having lowest risk).

In order to produce children we consider all pairs of portfolios, one portfolio from Q_1, the other from Q_2, so q^2 parent portfolio pairs in total. For each parent portfolio pair a single child is produced using crossover. In our crossover procedure:

- If an asset is present in both of the parent portfolios it is present in the child (and so in S_{in}).
- If it is absent from both of the parent portfolios it is absent in the child (and so in S_{out}).
- If it is present in one of the parent portfolios (absent in the other) then its presence (or not) in the child will be decided as a result of optimisation.

Mutation is standard within GAs and introduces a degree of stochastic variation, typically with low probability. In the computational results presented below we ran our GA for four generations, with mutation occurring in just the third generation. In our GA a child (with probability 0.03) is mutated by randomly selecting one asset in the child portfolio and replacing it by a random asset not present in the child portfolio.

Each child (for which the sets S_{in} and S_{out} have been decided after crossover and mutation) is optimised by solving $F(S_{\text{in}}, S_{\text{out}})$. Note here that we cannot guarantee that we get a feasible solution when we solve this subset optimisation problem, i.e. it is possible that there is no feasible child given the choice that has been made of S_{in} and S_{out} via crossover and mutation.

In our GA to generate a new population we combine the P members of the current population with the set of feasible children, sort the portfolios in this combined set into increasing risk (variance) order and take the first P members of this ordered list to constitute the new population for the next generation. At the end of the GA the P portfolios in the final population contribute to the cardinality constrained efficient frontier (though note here that we do eliminate at this stage any portfolios that are dominated by others in the final population).

Pseudocode and a flowchart for our GA heuristic are given in Appendix A and Fig. 2.

4.4. Tabu search

Tabu search (TS) is a local search heuristic described by Glover (1986) that uses deterministic control to overcome local optima in hill climbing. The basic principle of TS is to continue the search whenever a local optimum is encountered by allowing non–improving *moves*. A non-improving move is one that worsens the objective function.

Tabus are used to try and prevent cycling when moving away from local optima through non-improving moves. They are stored in a 'short-term' memory referred to as the *tabu list*. Moves that are on the tabu list cannot be made and a move typically remains on the tabu list for a fixed number of iterations (the *tabu tenure*). Although it generally prohibits the repetition of previously visited configurations, especially if the tabu tenure is not very small, the basic tabu search mechanism cannot guarantee the absence of cycles.

One danger of making moves tabu is that can prohibit attractive moves, even when there is no danger of cycling. It is thus often necessary to include algorithmic devices that will allow one to make moves that are tabu. One such device is the *aspiration criteria* where a move is allowed, even if tabu, provided it leads to a better solution than encountered in the search process so far.

In TS the search continues until some termination criteria is satisfied (e.g. fixed number of iterations, CPU time, fixed number of iterations since the solution was last improved).

For a more comprehensive overview of tabu search see Burke and Kendall (2005), Aarts and Lenstra (2003), Gendreau (2003), Glover and Laguna (1993, 1997).

In the literature, as surveyed above, examples of the application of TS to the cardinality constrained portfolio optimisation problem can be found in Chang et al. (2000), Schaerf (2002), Ehrgott et al. (2004), Cura (2009).

4.5. A tabu search heuristic for the CCEF

In our TS heuristic, given the desired return of ρ, we first generate $P = 100$ different portfolios, as for our GA, and then select the portfolio with the lowest risk (variance) as the initial starting solution. Let S_{in} be the set of assets in this initial solution.

In our approach we have a candidate list C of assets that can be considered for inclusion in the current solution, and a tabu list T of assets that cannot be considered. Initialise C with the $N/3$ assets with the highest return (excluding assets in S_{in}). Initialise T with the assets in $\{1,\ldots,N\} - S_{\text{in}} \cup C$.

In our TS heuristic we, at each iteration, randomly select an asset i in the current portfolio S_{in} and replace it by an randomly selected asset j in the candidate list C. Then we solve the subset optimisation problem F $(S_{\text{in}},S_{\text{out}})$ with $S_{\text{out}} = \{1,\ldots,N\} - S_{\text{in}}$. If the portfolio resulting from this optimisation is better (of lower risk) than the current solution then it replaces the current solution and asset i is added to the tabu list T. If the portfolio resulting from this optimisation is not better than the current solution then asset j is added to the tabu list T. The candidate list is then updated by adding assets from the tabu list that are no longer tabu. We terminate our TS heuristic after a fixed number of iterations and use a tabu tenure of seven iterations.

Pseudocode and a flowchart for our TS heuristic are given in Appendix A and Fig. 3.

4.6. Simulated annealing

Simulated annealing (SA) is a local search heuristic first used for optimisation by Kirkpatrick et al. (1983) and Cerny (1985). It begins with a single starting solution and explores potential moves (as does TS). In SA moves to worse solutions are accepted with a specified probability that decreases over the course of the algorithm. This probability is related to a factor known as *temperature*.

For a more comprehensive overview of simulated annealing see Burke and Kendall (2005), Aarts and Lenstra (2003).

In the literature, as surveyed above, examples of the application of SA to the cardinality constrained portfolio optimisation problem can be found in Chang et al. (2000), Schaerf (2002), Crama and Schyns (2003), Derigs and Nickel (2003, 2004), Maringer and Kellerer (2003), Ehrgott et al. (2004), Cura (2009), Ruiz-Torrubiano and Suarez (2010).

4.7. A simulated annealing heuristic for the CCEF

In our SA heuristic, given the desired return of ρ, we generate an initial starting solution (a set S_{in} of assets in the portfolio) in the same manner as in our TS heuristic above.

At each iteration we randomly select an asset i in the current solution S_{in} and swap it with a randomly selected asset j not in the current solution (so $j \notin S_{\text{in}}$) to give a new set $S_{\text{in}} = S_{in} \cup [j] - [i]$. Then we solve the subset optimisation problem F $(S_{\text{in}},S_{\text{out}})$ with $S_{\text{out}} = \{1,\ldots, N\} - S_{\text{in}}$. If the portfolio resulting from this optimisation is better (of lower risk) than the current solution then it replaces the current solution. If it is worse than the current solution then it is accepted (so replacing the current solution) with probability $e^{-(\textit{difference in solution risk values})/(\textit{current temperature})}$. The current temperature is reduced by a constant (cooling) factor at each iteration.

We terminate our SA heuristic after a fixed number of iterations. In the computational results given later below we use a cooling factor of 0.95 and an initial temperature derived from the objective function value of the initial starting solution.

Pseudocode and a flowchart for our SA heuristic are given in Appendix A and Fig. 4.

5. Computational results

We tested the performance of our GA, TS and SA metaheuristics for finding the cardinality constrained efficient frontier using publicly available test problems relating to seven major market indices, available from OR-Library (Beasley, 1990).

Five of our market indices were the Hang Seng (Hong Kong), DAX 100 (Germany), FTSE 100 (UK), S&P 100 (USA) and the Nikkei 225 (Japan), as taken from: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html. All of these problems were considered previously by Chang et al. (2000). The remaining two market indices were the S&P 500 (USA) and Russell 2000 (USA), as taken from: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/indtrackinfo.html. The size of our seven test problems ranged from $N = 31$ (Hang Seng) to $N = 1318$ (Russell 2000). We used $l_i = 0.01$, $u_i = 1$ $(i = 1,\ldots,N)$ and $K = 10$.

As we are interested in the cardinality constrained efficient frontier our results below are for tracing out this frontier using 50 equally spaced desired return levels ρ, see Eq. (2), between the return level associated with the minimum variance unconstrained portfolio and the return level associated with the maximum asset return $\max[\mu_i | i = 1,\ldots,N]$.

Our metaheuristics were implemented using AMPL and its associated script language. The solver we used was CPLEX 11.0. The system runs under Windows NT and in our computational work we used an Intel Core2 pc with a 2.40 GHz processor and 3.24 GB RAM.

5.1. CPLEX results

Before using the metaheuristic approaches presented above to solve for the CCEF we investigated using CPLEX to test how effectively it could determine CCEFs. Potentially, for example, should CPLEX be able to optimally solve for the CCEF, i.e. to optimally solve the CCEF QMIP (Eqs. (1)–(6)), there may be no need for any metaheuristic approaches.

As stated previously above our focus in this paper is on the problem as defined by Chang et al. (2000), Eqs. (1)–(6), where we seek precisely K assets in the portfolio, so $\sum_{i=1}^{N}\delta_i = K$ as Eq. (5). However, because a number of authors in the literature have considered the variant of the problem where the equality in Eq. (5) is replaced by inequality, so $\sum_{i=1}^{N}\delta_i \leqslant K$, we also consider here how CPLEX performs on this variant of the problem.

We tested CPLEX (version 11.0) on one of the smaller test problems (DAX 100, N = 85 assets) and the results are shown in Table 2. As mentioned above these results are for 50 equally spaced return levels. So for example in this table we have that for the DAX 100 with K = 5 and equality in terms of the number of chosen assets, it required 58336 s (over 16 h) to trace out the CCEF over the 50 equally spaced return levels.

It is clear from Table 2 that the inequality case (for the DAX 100 at least) is computationally far easier than the equality case. We also attempted to solve the largest test problem (Russell 2000, N = 1318 assets) for the same set of eight cases (K = 2, 3, 4, 5 and equality/inequality) as we considered for the DAX 100 (Table 2). CPLEX was unable to solve even a single one of these eight cases (not even K = 2, inequality) within a time limit of 7200 s (2 h).

Based on Table 2 (as well as other computational experimentation not reported in detail here) we would conclude that solving the CCEF QMIP using CPLEX is not a computationally effective approach. As such we are justified in adopting metaheuristic approaches to the problem. Note here that these results for CPLEX accord with other results presented in the literature (Shaw et al., 2008; Bertsimas and Shioda, 2009), albeit those results relate to an earlier version of CPLEX.

5.2. Parameter values

In our heuristics as presented above we have assigned values to parameters (e.g. population size in our GA). Readers familiar with work involving heuristic algorithms will know that often in such algorithms parameter values need to be assigned (indeed they are often an integral part of the general solution approach that is being particularised for the problem at hand). In this section we outline how we assigned such values. This assignment was arrived at by exploring a limited number of different parameter values using just one of our seven data sets, the smallest of these sets, namely the Hang Seng with N = 31 assets.

In general in measuring the quality of a heuristic one would like to measure the deviation of the heuristic solution from the optimal solution. However for the CCEF, as the results in Table 2 illustrate, the optimal frontier is typically unknown. As such in measuring the quality of the results produced by our heuristics we adopt the same approach as used previously by Chang et al. (2000). This involves calculating the percentage deviation of points on the heuristically calculated CCEF from the unconstrained efficient frontier (the UEF, which can be easily calculated using QP). Readers interested in precise details as to how these percentage deviations (errors) are calculated can find them in Chang et al. (2000).

5.2.1. GA parameter values

Within our GA we need to decide parameter values for population size, parental set size and mutation probability.

We began by varying population size. Alander (1992) suggested that a population size around 50–200 is suitable for most problems, thus we tested P = 50, 100, 150. Table 3 gives the results obtained. In that table we show the mean and median percentage errors as well as the computation time in seconds for 50 equally spaced desired return levels when our GA is applied to our chosen data set.

Given the results in Table 3 for our chosen values of P it is clear P = 150 offers no advantages, being worse on all three measures than P = 50,100. We decided to use P = 100 since it gives error results effectively equivalent to P = 50, but offers greater opportunities for exploration of the search space.

With the population size determined (i.e. working in a sequential fashion to decide parameter values) we next considered parental set size. Here we tried values of 3, 5 and 7. Based on the results shown in Table 3 we decided to use a parental set size of 5 (being influenced by the low median error associated with this value, and the opportunity for more exploration of the search space afforded by higher parental set size values).

Our final decision for the GA was for mutation probability. Mutation in GAs is typically assigned a low value. Here we tried values of 0.01, 0.03 and 0.05. Based on the results shown in Table 3 we decided to use a mutation probability of 0.03 (involving only slightly more time than 0.01, and with a lower median error).

5.2.2. TS parameter values

In our TS heuristic we need a value for tabu tenure. Glover and Laguna (1993) suggested a minimum tabu tenure of 7. For this parameter we tested three values: 5, 7 and 10; with the results being seen in Table 4. Based on these results we decided to use a tabu tenure of 7 (being influenced by the lower median error).

Table 2
Computation time (seconds) for the DAX 100 CCEF using CPLEX.

	K = 2	K = 3	K = 4	K = 5
Equality case (precisely K assets in the portfolio)	62	527	6984	58336
Inequality case ($\leqslant K$ assets in the portfolio)	19	50	106	138

Table 4
TS parameter results.

Percentage error and time	Tabu tenure		
	5	7	10
Mean	0.7645	0.8234	1.1529
Median	0.4173	0.3949	0.5169
Time (s)	69	76	84

Table 3
GA parameter results.

Percentage error and time	Population size			Parental set size			Mutation probability		
	50	100	150	3	5	7	0.01	0.03	0.05
Mean	0.8496	0.8501	0.9100	0.6959	0.8501	0.7668	0.8197	0.8501	0.9089
Median	0.5989	0.5873	0.6105	0.6104	0.5873	0.7457	0.6103	0.5873	0.5873
Time (s)	64	76	112	47	76	124	67	76	101

5.2.3. SA parameter values

In our SA heuristic the parameter we need to decide is the cooling schedule. The typical range for this value is between 0.90 and 0.99. For this parameter we tested three values: 0.90, 0.95, 0.975; with the results being seen in Table 5. Based on these results we decided to use a value of 0.95 as it dominated the other two values seen with regard to both mean and median error.

Table 5
SA parameter results.

Percentage error and time	Cooling schedule		
	0.90	0.95	0.975
Mean	1.5806	1.0589	1.0913
Median	1.5791	0.5355	0.9094
Time (s)	67	76	86

5.3. Heuristic results

The computational results reported in this paper examine 50 different return levels. With regard to the number of iterations, which is the termination criteria for both our TS and SA heuristics, we used 100 iterations at each return level for the TS heuristic and 50 iterations at each return level for the SA heuristic.

In Table 6 we show for each of our data sets and each of our heuristics: the mean, median, minimum and maximum percentage errors as well as the computation time in seconds.

Considering our GA, TS and SA heuristics as presented in this paper, labeled (Woodside-Oriakhi et al. in Table 6), we would make the following points with regard to the average values over the seven test problems:

- SA is not competitive with GA and TS, having higher mean and median errors and a higher computation time.

Table 6
Test problem results.

Index	N	Percentage error and time	Genetic algorithm		Tabu search		Simulated annealing	
			Chang et al.	Woodside-Oriakhi et al.	Chang et al.	Woodside-Oriakhi et al.	Chang et al.	Woodside-Oriakhi et al.
Hang Seng		Mean	0.9457	0.8501	0.9908	0.8234	0.9892	1.0589
		Median	1.1819	0.5873	1.1992	0.3949	1.2082	0.5355
	31	Minimum		0.0036		0.0068		0.0349
		Maximum		2.9034		4.6096		4.6397
		Time (s)	172	76	74	85	79	99
DAX 100		Mean	1.9515	0.7740	3.0635	0.7190	2.4299	1.0267
		Median	2.1262	0.2400	2.5383	0.4298	2.4675	0.8682
	85	Minimum		0.0000		0.0149		0.0278
		Maximum		4.6811		2.7770		4.4123
		Time (s)	544	74	199	113	210	293
FTSE 100		Mean	0.8784	0.1620	1.3908	0.3930	1.1341	0.8952
		Median	0.5938	0.0820	0.6361	0.2061	0.7137	0.3944
	89	Minimum		0.0000		0.0019		0.0230
		Maximum		0.7210		3.4570		10.2029
		Time (s)	573	95	246	232	215	286
S&P 100		Mean	1.7157	0.2922	3.1678	1.0358	2.6970	3.0952
		Median	1.1447	0.1809	1.1487	1.0248	1.1288	2.1064
	98	Minimum		0.0007		0.0407		0.8658
		Maximum		1.6295		3.0061		8.6652
		Time (s)	638	100	225	222	242	371
Nikkei 225		Mean	0.6431	0.3353	0.8981	0.7838	0.6370	1.1193
		Median	0.6062	0.3040	0.5914	0.6526	0.6292	0.6877
	225	Minimum		0.0180		0.0085		0.0113
		Maximum		1.0557		2.6082		3.9678
		Time (s)	1964	104	545	414	553	604
Average Chang et al. problems		Mean	1.2269	0.4827	1.9022	0.7510	1.5774	1.4391
		Median	1.1306	0.2788	1.2227	0.5416	1.2295	0.9184
		Minimum		0.0045		0.0146		0.1926
		Maximum		2.1981		3.2916		6.3776
		Time (s)	778	90	258	213	260	331
S&P 500		Mean		2.0205		1.4689		5.2502
		Median		0.1899		1.1047		4.5142
	457	Minimum		0.0114		0.0335		0.1552
		Maximum		21.1701		5.1203		13.9470
		Time (s)		187		660		719
Russell 2000		Mean		4.7797		0.7345		4.1102
		Median		0.0940		0.2700		3.8136
	1318	Minimum		0.0001		0.0097		0.0001
		Maximum		58.7478		3.8205		8.5477
		Time (s)		239		729		868
Average all problems		Mean		1.3163		0.8512		2.3651
		Median		0.2397		0.5833		1.8457
		Minimum		0.0048		0.0166		0.1597
		Maximum		12.9869		3.6284		7.7689
		Time (s)		125		351		463

Computation times for the work of Chang et al. (2000) as shown above should be divided by a factor of 70 to be comparable with the hardware we have used.

- TS has a lower mean error, but a higher median error, than GA, and takes more computation time.

Comparing mean errors for GA and TS over the seven test problems individually we have that for three problems GA is better than TS, for four problems TS is better than GA. Considering median errors we have that for all problems except the Hang Seng GA is better than TS.

For all of our heuristics the computation time required is not excessive, the largest computation time seen in Table 6 being 868 s, approximately 15 min.

Also presented in Table 6 are the mean and median percentage errors and computation times for the five smaller test problems as given in Chang et al. (2000) using their GA, TS and SA heuristics, henceforth denoted by GA-Chang, TS-Chang and SA-Chang.

Comparing, for these five smaller test problems, our results with the results of Chang et al. (2000) we would make the following points:

- Our GA dominates GA-Chang since for all five test problems our GA gives both a lower mean error and a lower median error. Moreover our GA mean error is lower than that of GA-Chang by a factor of 1.2269/0.4827 = 2.5; our GA median error is lower than that of GA-Chang by a factor of 1.1306/0.2788 = 4.1.
- Our TS heuristic effectively dominates TS-Chang since for four of the five test problems our TS heuristic gives a lower mean and median error than TS-Chang. Our TS mean error is lower than that of TS-Chang by a factor of 1.9022/0.7510 = 2.5; our TS median error is lower than that of TS-Chang by a factor of 1.2227/0.5416 = 2.3.
- Our SA heuristic is broadly equivalent to SA-Chang with, on average, a slightly lower mean and median error, but only dominating SA-Chang (in terms of better mean and median errors) for two of the five problems.

With regard to computation time the times given for the work of Chang et al. (2000) relate to different hardware than we have used. Utilising Dongarra (2009) it is possible to make an *approximate* estimate of the relative speed of the hardware involved. On this basis the computation times for the work of Chang et al. (2000) as shown in Table 6 should be divided by a factor of 70 to be comparable with the hardware we have used. As such we can conclude that for these smaller test problems our GA and TS heuristics take longer, but give better quality results in a reasonable time (an average of 1.5 min for our GA; 3.6 min for our TS), than GA-Chang and TS-Chang.

As we have a number of results from different heuristic approaches we can pool results, i.e. combine together the efficient portfolios from each of the heuristics and eliminate any portfolios that are dominated. In Table 7 we show the pooled results as given in Chang et al. (2000) and present the pooled results for our three heuristics.

In that table we use the symbol: to denote pooling so, for example, TA:SA denotes pooling the results from our TS and SA algorithms together. Note here that whilst we are able in Table 7 to give four sets of pooled results for our heuristics, namely {GA:TS:SA; GA:TS; GA:SA; TS:SA}, we are only able to give one set of pooled results {GA-Chang:TS-Chang:SA-Chang} for Chang et al. (2000) as they do not give separate pooled results for {GA-Chang:TS-Chang; GA-Chang:SA-Chang; TS-Chang:SA-Chang} in their paper.

Table 7
Pooled results.

Index	*N*	Percentage error and time	Pooled heuristics				
			Chang et al.	Woodside-Oriakhi et al.			
			GA-Chang:TS-Chang:SA-Chang	GA:TS:SA	GA:TS	GA:SA	TS:SA
Hang Seng	31	Mean	0.9332	0.4265	0.4098	0.6404	0.6036
		Median	1.1899	0.1839	0.1948	0.3669	0.3745
		Time (s)	325	260	161	175	184
DAX 100	85	Mean	2.1927	0.6539	0.4696	0.7055	0.7070
		Median	2.4626	0.2194	0.2073	0.2275	0.4247
		Time (s)	953	480	187	367	406
FTSE 100	89	Mean	0.7790	0.4418	0.2690	0.1598	0.5284
		Median	0.5960	0.1074	0.0851	0.0935	0.2061
		Time (s)	1034	613	327	381	518
S&P 100	98	Mean	1.3106	0.6748	0.5109	0.6172	1.0944
		Median	1.0686	0.2395	0.2756	0.2712	1.0495
		Time (s)	1105	693	322	471	593
Nikkei 225	225	Mean	0.5690	0.7307	0.7214	0.3870	0.9119
		Median	0.5844	0.3223	0.3223	0.2785	0.5481
		Time (s)	3062	1122	518	708	1018
Average, Chang et al. problems		Mean	1.1569	0.5855	0.4761	0.5020	0.7691
		Median	1.1803	0.2145	0.2170	0.2475	0.5206
		Time (s)	1296	634	303	420	544
S&P 500	457	Mean		0.8385	0.7549	1.1319	1.5500
		Median		0.2861	0.2685	0.2181	1.1581
		Time (s)		1566	847	906	1379
Russell 2000	1318	Mean		0.7192	0.7192	4.7797	0.8355
		Median		0.1039	0.1040	0.0940	0.2890
		Time (s)		1836	968	1107	1597
Average, all problems		Mean		0.6408	0.5507	1.2031	0.8901
		Median		0.2089	0.2082	0.2214	0.5786
		Time (s)		939	476	588	814

Computation times for the work of Chang et al. (2000) as shown above should be divided by a factor of 70 to be comparable with the hardware we have used.

Comparing the average values over all seven test problems presented in Table 7 it seems clear that there is, on average, little advantage to including results from our SA heuristic in pooling. Rather the best pooled results (both in terms of mean and median errors, and in terms of computation time) come from pooling our GA and TS heuristics. Note here however that if we look at the individual test problem results we can see that there is sometimes an advantage gained from including results from our SA heuristic in pooling (e.g. for the FTSE 100 pooling the GA and SA results gives a lower mean error than pooling the GA and TS results).

Comparing (pooled) results for GA:TS in Table 7 with the individual results for GA and TS as presented in Table 6 it seems clear that the quality of results are improved considerably by pooling. For example the average mean error for GA:TS is 0.5507%, whereas our GA and TS heuristics individually have mean errors of 1.3163% and 0.8512%, respectively.

For the five smaller test problems the results for GA:TS are of much better quality than the pooled results for all three of the Chang et al. (2000) heuristics, GA-Chang:TS-Chang:SA-Chang. Specifically:

- The mean error for GA:TS is 0.4761%, the mean error for GA-Chang:TS-Chang:SA-Chang is 1.1569%, a factor of 1.1569/0.4761 = 2.4 better.

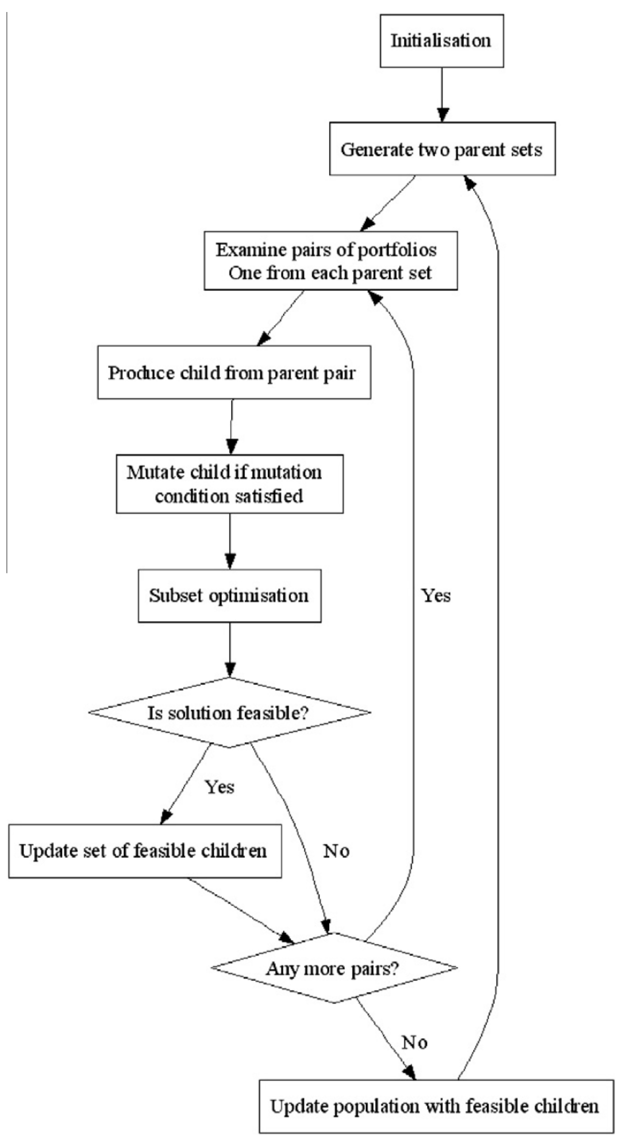

Fig. 2. GA heuristic algorithm flowchart.

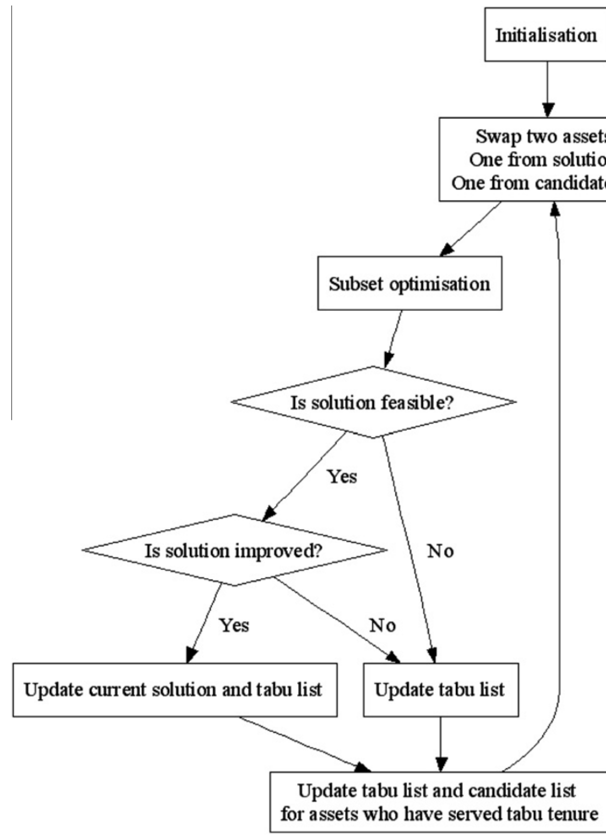

Fig. 3. TS heuristic algorithm flowchart.

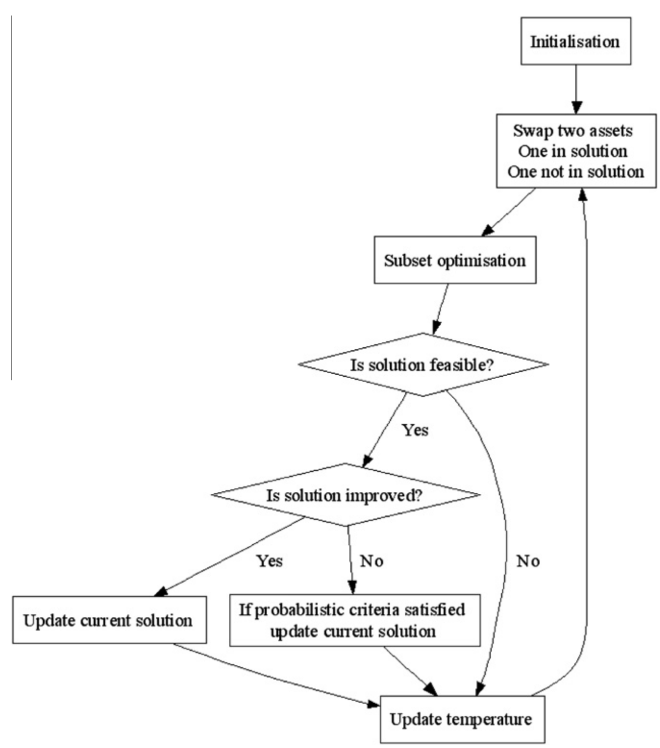

Fig. 4. SA heuristic algorithm flowchart.

- The median error for GA:TS is 0.2170%, the median error for GA-Chang:TS-Chang:SA-Chang is 1.1803%, a factor of 1.1803/0.2170 = 5.4 better.
- For four of the five test problems GA:TS has a better mean error than the pooled Chang heuristics, GA-Chang:TS-Chang:SA-Chang.
- For all five test problems GA:TS has a better median error than the pooled Chang heuristics, GA-Chang:TS-Chang:SA-Chang.

Based upon the detailed discussion given above we would conclude that our heuristics give better quality solutions than previous heuristics presented in the literature, albeit at the expense of more computation time.

6. Conclusion

In this paper we have presented three metaheuristic algorithms (based upon genetic algorithms, tabu search and simulated annealing) to find the cardinality constrained efficient frontier that arises in financial portfolio optimisation. Computational results were presented for our heuristics on test problems considered previously in the literature, as well as on larger test problems involving up to 1318 assets.

Our results indicate that our heuristics give better quality solutions than previous heuristics presented in the literature, albeit at the expense of more computation time. However, in all cases, our computation times were reasonable and were never more than fifteen minutes on a modern pc, even for the largest problem.

A feature of our metaheuristics is that we have a subset optimisation step so that we solve, to optimality, a (small) mixed-integer quadratic optimisation problem. We believe that the quality of our results indicates that this can be a useful strategy to employ within the context of cardinality constrained portfolio optimisation.

Appendix A

In this appendix we present pseudocode for our three heuristic algorithms (see Figs. 2–4).

Algorithm 1: GA heuristic algorithm psuedocode

R_{min} is the return level associated with the minimum variance unconstrained portfolio
R_{max} is the maximum expected return for all assets, thus $R_{max} = \max[\mu_i|i = 1,\ldots,N]$
O_{xy} is the child of $x \in Q_1$ and $y \in Q_2$ after subset optimisation
O^* is the set of feasible children
P^* is the set of current population members
G is the number of generations
Γ is the set of all assets $[1,\ldots,N]$

```
begin
for ρ := R_min,...,R_max do                 /examine values for ρ equally spaced in [R_min,R_max]/
initialise P*                               /random initialisation, S_in = max[2K,20] assets/
determine S_out := Γ − S_in ∀p ∈ P*
solve F (S_in,S_out) ∀p ∈ P*                /subset optimisation/
   for g := 1,...,G do                      /G generations in all/
   O* := ∅
   select Q_1, Q_2 by selection criteria    /parent sets/
      for all x ∈ Q_1 and y ∈ Q_2 do        /crossover to produce children/
```

Algorithm 1 (*continued*)

Algorithm 1: GA heuristic algorithm psuedocode

```
      S_in := x ∩ y
      S_out := (Γ − x) ∩ (Γ − y)
         if g := g* then                    /mutation at generation g*/
         if mutation probability then
               for i ∈ S_in and j ∈ S_out do
               S_in := S_in ∪ [j] − [i]
               S_out := S_out ∪ [i] − [j]
               end for
         end if
         end if
      solve F (S_in,S_out)                  /subset optimisation/
         if F (S_in,S_out) is feasible then /evaluate solution/
               O* := O* ∪ O_xy              /collect feasible children/
         end if
      end for
      P* := P* ∪ O* and sort by variance    /combine children with current population/
      P* := first P in P*                   /new population/
   end for
end for
end
```

Algorithm 2: TS heuristic algorithm psuedocode

R_{min} is the return level associated with the minimum variance unconstrained portfolio
R_{max} is the maximum expected return for all assets, thus $R_{max} = \max[\mu_i|i = 1,\ldots,N]$
P^* is the initial set of solutions
G is the number of iterations
Γ is the set of all assets $[1,\ldots,N]$
S^* is the set of assets in the current solution

```
begin
for ρ := R_min,...,R_max do                 /examine values for ρ equally spaced in [R_min,R_max]/
initialise P*                               /random initialisation, S_in = max[2K,20] assets/
determine S_out := Γ − S_in ∀p ∈ P*
solve F (S_in,S_out) ∀p ∈ P*                /subset optimisation/
S* := {assets in p ∈ P*|p has minimum variance}   /initial solution/
initialise C := {the N/3 assets with highest return excluding assets in S*}
initialise T := Γ − S* ∪ C
 for g := 1,...,G do                        /G iterations in all/
 randomly select i ∈ S* and j ∈ C
 S_in := S* ∪ [j] − [i]                     /neighbourhood solution/
 S_out := Γ − S_in
 solve F (S_in,S_out)                       /subset optimisation/
    if F (S_in,S_out) is feasible then      /evaluate solution/
       if var (solution) < var
```

Algorithm 2 (*continued*)

Algorithm 2: TS heuristic algorithm psuedocode

```
  (S*) then
        S* := S_in                      /improved solution/
        T := T ∪ [i]                    /update tabu list/
      else
        T := T ∪ [j]                    /update tabu list/
      end if
    else
      T := T ∪ [j]                      /update tabu list/
    end if
    check T and update C and T          /move assets who have served tabu tenure from T into C/
  end for
end for
end
```

Algorithm 3: SA heuristic algorithm psuedocode

$R_{\min}$ is the return level associated with the minimum variance unconstrained portfolio
$R_{\max}$ is the maximum expected return for all assets, thus $R_{\max} = \max[\mu_i | i = 1,\ldots,N]$
P^* is the initial set of solutions
G is the number of iterations
Γ is the set of all assets $[1,\ldots,N]$
S^* is the set of assets in the current solution
β is the current temperature
α is the cooling factor

```
begin
for ρ := R_min,...,R_max do             /examine values for ρ equally spaced in [R_min,R_max]/
initialise P*                           /random initialisation, S_in = max[2K,20] assets/
determine S_out := Γ − S_in ∀p ∈ P*
solve F (S_in,S_out) ∀p ∈ P*            /subset optimisation/
S* := {assets in p ∈ P*|p has minimum variance}   /initial solution/
β := var (S*)/10                        /initialise SA parameters/
α := 0.95
  for g := 1,...,G do                   /G iterations in all/
  randomly select i ∈ S* and j ∉ S*
  S_in := S* ∪ [j] − [i]
  S_out := Γ − S_in
  solve F (S_in,S_out)                  /subset optimisation/
    if F (S_in,S_out) is feasible then  /evaluate solution/
      if var (solution) < var (S*) then
          S* := S_in                    /improved solution/
      else
          r := a random number from [0,1]
          R := e^{−(var(solution)−var(S*))/β}
          if R > r then                 /criteria for accepting worse portfolio/
            S* := S_in
```

Algorithm 3 (*continued*)

Algorithm 3: SA heuristic algorithm psuedocode

```
          end if
        end if
      end if
    β := αβ                             /update temperature/
  end for
end for
end
```

References

Aarts, E.H.L., Lenstra, J.K. (Eds.), 2003. Local Search in Combinatorial Optimization. Princeton University Press, Princeton, USA.

Alander, J.T., 1992. On optimal population size of genetic algorithms. In: Computer Systems and Software Engineering – CompEuro 1992 Proceedings. IEEE Computer Society Press, pp. 65–70.

Anagnostopoulos, K.P., Mamanis, G., 2010. A portfolio optimization model with three objectives and discrete variables. Computers & Operations Research 37, 1285–1297.

Beasley, J.E., 1990. OR-Library: Distributing test problems by electronic mail. Journal of the Operational Research Society 41, 1069–1072.

Beasley, J.E., 2002. Population heuristics. In: Pardalos, P.M., Resende, M.G.C. (Eds.), Handbook of Applied Optimization. Oxford University Press, Oxford, pp. 138–157.

Bertsimas, D., Shioda, R., 2009. Algorithm for cardinality-constrained quadratic optimization. Computational Optimization and Applications 43, 1–22.

Bienstock, D., 1996. Computational study of a family of mixed-integer quadratic programming problems. Mathematical Programming 74, 121–140.

Bonami, P., Lejeune, M.A., 2009. An exact solution approach for portfolio optimization problems under stochastic and integer constraints. Operations Research 57, 650–670.

Branke, J., Scheckenbach, B., Stein, M., Deb, K., Schmeck, H., 2009. Portfolio optimization with an envelope-based multi-objective evolutionary algorithm. European Journal of Operational Research 199, 684–693.

Burke, E.K., Kendall, G. (Eds.), 2005. Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques. Springer.

Cerny, V., 1985. Thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Applications 45, 41–51.

Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M., 2000. Heuristics for cardinality constrained portfolio optimisation. Computers & Operations Research 27, 1271–1302.

Chang, T.-J., Yang, S.-C., Chang, K.-J., 2009. Portfolio optimization problems in different risk measures using genetic algorithm. Expert Systems with Applications 36, 10529–10537.

Chiam, S.C., Tan, K.C., Al Mamum, A., 2008. Evolutionary multi-objective portfolio optimization in practical context. International Journal of Automation and Computing 5, 67–80.

Corazza, M., Favaretto, D., 2007. On the existence of solutions to the quadratic mixed-integer mean-variance portfolio selection problem. European Journal of Operational Research 176, 1947–1960.

Crama, Y., Schyns, M., 2003. Simulated annealing for complex portfolio selection problems. European Journal of Operational Research 150, 546–571.

Cura, T., 2009. Particle swarm optimization approach to portfolio optimization. Nonlinear Analysis: Real World Applications 10, 2396–2406.

Derigs, U., Nickel, N.-H., 2003. Meta-heuristic based decision support for portfolio optimisation with a case study on tracking error minimization in passive portfolio management. OR Spectrum 25, 345–378.

Derigs, U., Nickel, N.-H., 2004. On a local-search heuristic for a class of tracking error minimization problems in portfolio management. Annals of Operation Research 131, 45–77.

Dongarra, J.J., 2009. Performance of Various Computers Using Standard Linear Equations Software, Report CS-89-85. University of Tennessee, USA. Available from: <http://www.netlib.org/benchmark/performance.ps> (accessed 07.02.10).

Duran, F.C., Cotta, C., Fernandez, A.J., 2009. Evolutionary optimization for multiobjective portfolio selection under Markowitz's model with application to the Caracas stock exchange. In: Chiong, R. (Ed.), Nature-Inspired Algorithms for Optimisation: Studies in Computational Intelligence, vol. 193. Springer, Berlin/Heidelberg, pp. 489–509.

Ehrgott, M., Klamroth, K., Schwehm, C., 2004. An MCDM approach to portfolio optimization. European Journal of Operational Research 155, 752–770.

Ellison, E.F.D., Hajian, M., Levkovitz, K., Maros, I., Mitra, G., 1999. A Fortran Based Mathematical Programming System. FortMP, Brunel University, UK; NAG Ltd., Oxford, UK.

Fernandez, A., Gomez, S., 2007. Portfolio selection using neural networks. Computers & Operations Research 34, 1177–1191.

Fourer, F., Gay, D.M., Kernighan, B.W., 2002. AMPL: A Modeling Language for Mathematical Programming. Brooks/Cole Publishing Company/Cengage Learning.
Gendreau, M., 2003. An introduction to tabu search. In: Glover, F., Kochenberger, G.A. (Eds.), Handbook of Metaheuristics. Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 37–54.
Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Computers & Operations Research 13, 533–549.
Glover, F., Laguna, M., 1993. Tabu search. In: Reeves, C.R. (Ed.), Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publications, Oxford, UK, pp. 70–150.
Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers, Dordrecht, The Netherlands.
Gulpinar, N., An, L.T.H., Moeini, M., 2010. Robust investment strategies with discrete asset choice constraints using DC programming. Optimization 59, 45–62.
Hillier, F.S., Lieberman, G.J., 2010. Introduction to Operations Research, ninth ed. McGraw-Hill.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence. University of Michigan Press, Ann Arbor, MI, USA.
Jobst, N.J., Horniman, M.D., Lucas, C.A., Mitra, G., 2001. Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints. Quantitative Finance 1, 489–501.
Kellerer, H., Mansini, R., Speranza, M.G., 2000. On selecting portfolios with fixed costs and minimum transaction lots. Annals of Operations Research 99, 287–304.
Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220, 671–680.
Larranaga, P., Lozano, J.A. (Eds.), 2001. Estimation of Distribution Algorithms: A New Tool For Evolutionary Computation. Springer.
Lemke, C.E., Howson, J.T., 1964. Equilibrium points of bimatrix games. Journal of the Society for Industrial and Applied Mathematics 12, 413–423.
Li, D., Sun, X., Wang, J., 2006. Optimal lot solution to cardinality constrained mean-variance formulation for portfolio selection. Mathematical Finance 16, 83–101.
Mansini, R., Speranza, M.G., 1999. Heuristic algorithms for the portfolio selection problem with minimum transaction lots. European Journal of Operational Research 114, 219–233.
Maringer, D., Kellerer, H., 2003. Optimization of cardinality constrained portfolios with a hybrid local search algorithm. OR Spectrum 25, 481–495.
Markowitz, H., 1952. Portfolio selection. Journal of Finance 7, 77–91.
Markowitz, H.M., 1956. The optimization of a quadratic function subject to linear constraints. Naval Research Logistics Quarterly 3, 111–133.
Mitchell, M., 1996. An Introduction to Genetic Algorithms. MIT Press, Cambridge, MA, USA.
Moral-Escudero, R., Ruiz-Torrubiano, R., Suarez, A., 2006. Selection of optimal investment portfolios with cardinality constraints. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation, pp. 2382–2388.
Pai, G.A.V., Michel, T., 2009. Evolutionary optimization of constrained k-means clustered assets for diversification in small portfolios. IEEE Transactions on Evolutionary Computation 13, 1030–1053.
Ruiz-Torrubiano, R., Suarez, A., 2010. Hybrid approaches and dimensionality reduction for portfolio selection with cardinality constraints. IEEE Computational Intelligence Magazine 5 (2), 92–107.
Schaerf, A., 2002. Local search techniques for constrained portfolio selection problems. Computational Economics 20, 177–190.
Shaw, D.X., Liu, S., Kopman, L., 2008. Lagrangian relaxation procedure for cardinality-constrained portfolio optimization. Optimization Methods & Software 23, 411–420.
Soleimani, H., Golmakani, H.R., Salimi, M.H., 2009. Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm. Expert Systems with Applications 36, 5058–5063.
Streichert, F., Tanaka-Yamawaki, M., 2006. The effect of local search on the constrained portfolio selection problem. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation, pp. 2368–2374.
Syam, S.S., 1998. A dual ascent method for the portfolio selection problem with multiple constraints and linked proposals. European Journal of Operational Research 108, 196–207.
Vielma, J.P., Ahmed, S., Nemhauser, G.L., 2008. A lifted linear programming branch-and-bound algorithm for mixed-integer conic quadratic programs. INFORMS Journal on Computing 20, 438–450.