Variable Neighborhood Search for Noisy Problems Applied to Project Portfolio Analysis

Walter J. Gutjahr, Stefan Katzensteiner, Peter Reiter

Dept. of Statistics and Decision Support Systems, University of Vienna
{walter.gutjahr,stefan.katzensteiner,peter.reiter}@univie.ac.at,
http://mailbox.univie.ac.at/walter.gutjahr/

Abstract. Motivated by an application in project portfolio analysis under uncertainty, we develop an algorithm S-VNS for solving stochastic combinatorial optimization (SCO) problems based on the Variable Neighborhood Search (VNS) metaheuristic, and show its theoretical soundness by a mathematical convergence result. S-VNS is the first general-purpose algorithm for SCO problems using VNS. It combines a classical VNS search strategy with a sampling approach with suitably increasing sample size. After the presentation of the algorithm, the considered application problem in project management, which combines a project portfolio decision on an upper level and project scheduling as well as staff assignment decisions on a lower level, is described. Uncertain work times require a treatment as an SCO problem. First experimental results on the application of S-VNS to this problem are outlined.

Keywords. Variable Neighborhood Search, stochastic combinatorial optimization, project portfolio selection, staff assignment, project scheduling.

1 Introduction

Stochastic combinatorial optimization (SCO) problems occur very frequently in diverse types of applications, since many optimization problems have a discrete or combinatorial structure, and very often, decisions have to be made under uncertainty. In this case, a standard approach offered by the operations research paradigm is to represent uncertain aspects quantitatively by a stochastic model, and to optimize the decision with respect to important characteristics (usually expected values, sometimes also variances, quantiles etc.) of the cost function which is then a random variable. For stochastic models involving a high degree of realism and therefore usually also a high degree of complexity, it can easily occur that these characteristics cannot be computed from explicit formulas or by means of numerical procedures, but must be estimated by simulation. This has led to the development of so-called *simulation-optimization* methods such as the Stochastic Branch-and-Bound Method [16], the Stochastic Ruler Method [2], or the Nested Partitions Method [17]. For a good survey on this area, cf. [5].

Especially in cases where already the underlying deterministic combinatorial optimization (CO) problem is computationally hard and practical problem instances are of a medium or large size, combined methods treating the combinatorial aspect of the problem by a *metaheuristic* seem to be advisable. Such metaheuristic-supported simulation-optimization algorithms have been developed, e.g., based on Simulated Annealing ([7], [8], [1]), Genetic Algorithms [4], or Ant Colony Optimization ([9], [10]).

Recently, much attention is given in the literature to the *Variable Neighborhood Search* (VNS) metaheuristic developed by Hansen and Mladenovich [12] which seems to produce excellent results on several hard deterministic CO problems. To our best knowledge, however, no attempt has been made up to now to develop an extension of a VNS algorithm for the treatment of SCO problems by a simulation-optimization approach. Thus, a general-purpose technique for treating SCO problems by a VNS-based method seems still to be lacking.

In the present work, we propose a VNS-based algorithm called S-VNS which is applicable to a broad class of SCO problems. We give a formal "proof of concept" for the algorithm by deriving a strict mathematical *convergence result*: Under mild conditions, the current solutions of S-VNS converge to (global) optimality. A convergence result for a metaheuristic is certainly not yet a sufficient condition for its practical usefulness. However, one would hesitate to apply a heuristic algorithm for which cases cannot be excluded where the optimal solution can *never* be found, no matter how much computation time is invested. In a deterministic context, convergence of the "best solution so far" to the optimum can typically be easily achieved by introducing a sufficient amount of randomness into the search process. In a stochastic context, however, the problem is complicated by two factors: First, the "best solution so far" can not be determined with certainty by simulation. Secondly, the question of a suitable sample size scheme in an iterative stochastic optimization procedure is non-trivial, as outlined very clearly by Homem-de-Mello in [13].

The development of our VNS variant for stochastic problems is mainly motivated by an application in the field of project portfolio analysis under uncertainty. Practical project management, especially in the R & D area, requires (repeated) decisions on sets of projects, so-called *portfolios*, that are to be carried out, while rejecting or deferring other project candidates because of capacity limits or of insufficient benefits to be expected from those projects (cf. , e.g., [15] or [6]). In real applications, this decision cannot be separated from two other decisions to be made on a "lower decision level": one on the scheduling of the required work packages of each project that has been chosen, the other on the assignment of personnel to work packages. The problem is stochastic in so far as the work times needed by each work package have to be considered as random variables. Finally, when considering objective functions, we do not only take account of economic goals and of potential tardiness of projects, but also include a strategic goal connected with the work of the staff in different "competencies". All in all, this gives a problem formulation that is sufficiently complex to serve as a test case for the proposed S-VNS algorithm.

The plan of the paper is the following: In section 2, the proposed algorithm S-VNS is presented. Section 3 contains the mathematical convergence result. In section 4, we present the application problem under consideration, outline the way it is attacked, and present results of the application of S-VNS for a specific example instance. Section 5 contains conclusions.

2 The Proposed Simulation-Optimization Algorithm

2.1 General Problem Structure

The proposed algorithm is designed to provide solutions to the following rather general type of stochastic combinatorial optimization problems:

$$\text{Minimize } f(x) = \mathrm{E}\,(F(x,\omega)) \quad \text{s.t.} \quad x \in S.$$

Therein, x is the decision, F is the cost function, ω denotes the influence of randomness, E denotes the mathematical expectation, and S is a set of feasible decisions, which we always assume as finite.

We focus on application cases where it is difficult or even impossible to compute $\mathrm{E}\,(F(x,\omega))$ numerically. Instead of determining approximations to this expected value by numerical techniques (e.g., numerical integration or analytical transform techniques), we resort to estimating it by sampling: For this purpose, a sample Z of s independent random *scenarios* $\omega_1, \ldots, \omega_s$ is drawn. The *sample average estimate* (SAE) is then given by $\tilde{f}(x) = (1/s) \sum_{\nu=1}^{s} F(x,\omega_\nu) \approx \mathrm{E}\,(F(x,\omega))$. It is easy to see that $\tilde{f}(x)$ is an unbiased estimator for $f(x)$.

A typical application of our approach is given by the case where the SAEs are obtained by Monte-Carlo simulation. In this situation, we can interpret a scenario ω_ν as a random number or a vector of random numbers allowing, together with the solution x and the cost function F, the computation of an observation $F(x,\omega_\nu)$.

2.2 The Algorithm S-VNS

Our proposed solution algorithm, S-VNS, follows the general lines of the VNS algorithm as described in [12], but extends it by the computation of SAEs at several places, and by the introduction of an additional step called *tournament*, where a current candidate solution x is compared to a solution $\hat{x}$ considered best so far, replacing $\hat{x}$ by x if the SAE of x (based on an appropriate sample size) turns out as better than the SAE of $\hat{x}$.

As ordinary VNS, the procedure S-VNS uses a hierarchy of *neighborhood structures*: For each $x \in S$, we assume a *k-neighborhood* $\mathcal{N}_k(x) \subseteq S$ of x as given, where $k = 0, \ldots, k_{max}$. It is assumed that $\mathcal{N}_0(x) = \{x\}$ for all $x \in S$. Furthermore, we assume that the sets $\mathcal{N}_k$ $(k = 0, \ldots, k_{max})$ are nonempty and pairwise disjoint. The last assumption comes in a natural manner if the neighborhoods are derived from a distance function $d(.,.)$ on S by the definition $\mathcal{N}_k(x) = \{y \in S \mid d(x,y) = k\}$. (An alternative assumption would be to consider neighborhoods that form a chain of inclusions: $\mathcal{N}_0(x) \subset \mathcal{N}_1(x) \subset \ldots \subset \mathcal{N}_{k_{max}}$. This situation occurs if one sets $\mathcal{N}_k(x) = \{y \in S \,|\, d(x,y) \leq k\}$. Our theoretical result can also be extended to this case, but we skip this point for the sake of brevity.)

Procedure S-VNS

```
set $m = 1$;
choose $x \in S$;
set $\hat{x} = x$; // best-so-far solution
repeat until stopping criterion is met{
  (1) set $k = 1$;
  (2) repeat until $k = k_{max}$ {
    (a) (shaking:) generate an $x' \in \mathcal{N}_k(x)$ at random;
    (b) (local search:)
        set $\bar{x} = x'$;
        set local-optimum-found = false;
        set $\rho = 1$;
        repeat until local-optimum-found or $\rho > \rho_{max}$ {
          draw a sample $Z$ of size $s_0$;
          compute the SAEs of $\bar{x}$ and of all solutions in $\mathcal{N}_1(\bar{x})$ w.r.t. $Z$;
          set $\bar{x}^* =$ solution in $\mathcal{N}_1(\bar{x})$ with best SAE;
          if ($\bar{x}^*$ has better SAE than $\bar{x}$) {
            set $\bar{x} = \bar{x}^*$;
            set $\rho = \rho + 1$;
          }
          else set local-optimum-found = true;
        }
        set $x'' = \bar{x}$;
    (c) (move-or-not:)
        draw a sample $Z'$ of size $s_0$;
        compute the SAEs of $x$ and of $x''$ w.r.t. $Z'$;
        if ($x''$ has better SAE than $x$) {
          set $x = x''$;
          set $k = 1$;
        }
        else set $k = k + 1$;
    (d) (tournament:)
        draw a sample $Z''$ of size $s_m$;
        compute the SAEs of $\hat{x}$ and of $x$ w.r.t. $Z''$;
        if ($x$ has better SAE than $\hat{x}$)
          set $\hat{x} = x$;
        set $m = m + 1$;
} }
```

Fig. 1. Pseudocode S-VNS.

Fig. 1 shows the pseudo-code of S-VNS. We call the part (a) – (d) of the execution starting with a shaking step and ending with a tournament step a *round* of the algorithm. The integer m ($m = 1, 2, \ldots$) is used as a round index. A round begins with the determination of a random neighbor solution x' to the current solution x ("shaking"), which aims at escaping from local optima. The size k of the neighborhood is dynamically controlled. The shaking mechanism must ensure that all solutions x' in the considered neighborhood have a positive probability of being selected. After that, x' is improved by local search steps until either a local optimum is found, or a certain predefined number ρ_{max} of steps is reached. Since the comparison of the current local search solution $\bar{x}$ with its neighbor solutions requires the estimation of objective function values, a sample Z (updated in each step) of constant size s_0 is used in each local search step.

Each round starts with some value of the neighborhood size variable k. We can group the rounds to *cycles*, where a new cycle is started any time when a round starts with $k = 1$. A cycle consists of one or several rounds with consecutively increasing values of k.

Local search results in a solution x''. In the step "move-or-not", this solution replaces the current incumbent x if it evaluates better at a further sample Z', again of size s_0. In the last case, the neighborhood size is reduced to one. Finally, the "tournament" between the incumbent x and the solution $\hat{x}$ considered best so far takes place. The size s_m of the sample Z'' on which this tournament is based has to be increased with suitable speed during the process in dependence on m. (This is contrary to the sizes of samples Z and Z' which are not increased.) We denote the solution x exposed to the tournament in round m by $x(m)$, and it will be shown in the next section that on certain conditions, the sequence $x(m)$ converges to a globally optimal solution.

3 Convergence Analysis

As it is seen from the pseudo-code, S-VNS works with different random samples at different occasions. Let us start their description with sample Z''. In round m, sample Z'' contains scenarios $\omega_1^m, \ldots, \omega_{s_m}^m$. We use the symbol ω^m for the s_m-tupel $(\omega_1^m, \ldots, \omega_{s_m}^m)$, and the symbol ω for $(\omega^1, \omega^2, \ldots)$.

Sample Z contains, in the ρ-th local search step of the m-th round, scenarios $\eta_{\rho,1}^m, \ldots, \eta_{\rho,s_0}^m$, and sample Z' contains scenarios which we denote by $\eta_{0,1}^m, \ldots, \eta_{0,s_0}^m$. By η^m, the matrix $(\eta_{\rho,\nu}^m)$ ($\rho = 0, \ldots, \rho_{max}$; $\nu = 1, \ldots, s_0$) is denoted, and $\eta = (\eta^1, \eta^2, \ldots)$ comprises all η^m.

Finally, there is an additional influence of randomness which is active in the shaking step of the algorithm. We denote the random scenario influencing the choice of the solution x' in the shaking step of round m by ξ^m, and set $\xi = (\xi^1, \xi^2, \ldots)$. Note that apart from ξ^m, the decision on x' is also dependent on the current x and on the current value of k.

The total information (ξ, η, ω) determining the random influence will be called *sample path*.

Throughout this article, we assume that all scenarios ω_ν^m, $\eta_{\rho,\nu}^m$ and ξ^m are independent. Moreover, it is assumed that the ω_ν^m have identical probability measures for all m and ν, the $\eta_{\rho,\nu}^m$ have identical probability measures for all m, ρ and ν, and the ξ^m have identical probability measures for all m. These assumptions are satisfied in a natural way if S-VNS applies Monte-Carlo simulation for cost evaluation, based on a series of independent random numbers the distribution of which is independent of x and m.

Note that we have two different probability mechanisms in the algorithm. The first mechanism generates the part (ξ, η) of the sample path. This mechanism completely determines the control flow in S-VNS, with the exception of the question whether in the tournament, $\hat{x}$ is updated to the value x or not. We denote the probability space consisting of the elements (ξ, η) by Ξ, and the probability measure on it by P. The second mechanism generates the part ω of the sample path. It influences the decision which solution wins a tournament. We denote the probability space consisting of the elements ω by Ω, and the probability measure on it by $\tilde{P}$.

For deriving our main theoretical result, we make use of the following lemma which has be proven by large-deviation theory:

Lemma 3.1 (Homem-de-Mello [13], Proposition 3.2). Suppose that for a scheme $(s_1, s_2, \ldots)$ of sample sizes and independent scenarios ω_ν^m,

(i) for each $x \in S$, the variances $var[F(x, \omega_1^m)]$ are bounded by some constant $M = M(x) > 0$,
(ii) the scenarios ω_ν^m have identical probability measures, and the SAEs $\tilde{f}_m(x) = (1/s_m) \sum_{\nu=1}^{s_m} F(x, \omega_\nu^m)$ are unbiased[1], i.e., $\mathrm{E}\,(\tilde{f}_m(x)) = f(x)$ for all x,
(iii) $\sum_{m=1}^{\infty} \alpha^{s_m} < \infty$ for all $\alpha \in]0, 1[$.

Then for each x, we have $\tilde{f}_m(x) \to f(x)$ $(m \to \infty)$ for $\tilde{P}$-almost all $\omega \in \Omega$.

We are now able to show convergence of S-VNS to a (globally) optimal solution. The proof is given in the Appendix.

Theorem 3.1. Consider S-VNS. Suppose that

(a) $\bigcup_{k=0}^{k_{max}} \mathcal{N}_k(x) = S$ for all $x \in S$,
(b) the assumptions (i) – (iii) of Lemma 3.1 are satisfied for the tournaments in S-VNS, and
(c) the distribution of $F(x, \omega)$ has a nonzero density in each point of $\mathrm{I\!R}$.

Then, for P-almost all $(\xi, \eta) \in \Xi$ and $\tilde{P}$-almost all $\omega \in \Omega$, there exists an integer m_0 (depending on the sample path (ξ, η, ω)) such that $x_m \in S^*$ for all $m \geq m_0$, where S^* is the set of optimal solutions.

Remark 1. In typical VNS applications, k_{max} is usually set to a smaller value than the minimum value ensuring that each solution can be reached from any x

[1] Homem-de-Mello also refers to a more general framework where $\mathrm{E}\,(F(x, \omega_\nu^m))$ can be different from $f(x)$. In our context, unbiasedness follows by definition.

within a k-neighborhood with $k \leq k_{max}$, as assumed in condition (a) of Theorem 3.1. In future investigations, one might try to generalize the result by omitting condition (a).

Remark 2. A sufficient condition for assumption (iii) in Lemma 2.1 to be satisfied is that the sample size s_m grows as $c_0 \cdot \sqrt{m}$ with a constant $c_0 > 0$, which is a fairly moderate growth. However, not all increasing samples size schedules satisfy assumption (iii); e.g., logarithmic growth is not sufficient.

4 The Application Problem: Stochastic Project Selection, Scheduling and Staffing

4.1 Problem Formulation

We apply the proposed algorithm to the stochastic version of a problem the deterministic version of which has been introduced in [11] (cf. also [15], [6]). The problem encompasses a project portfolio selection decision on an upper decision level and decisions on project scheduling and staff assignment on a lower level. The upper-level decision consists in the choice of a subset of projects from a given set of candidates $i = 1, \ldots, n$. Let the binary decision variable x_i take the value 1 iff project i is to be selected, and let $x = (x_1, \ldots, x_n)$. Projects have ready times ρ_i and due dates δ_i. The decision has to be made under uncertainty on the work times needed by the projects with respect to certain required human competencies $r = 1, \ldots, R$, which can be considered as "resources". It is assumed that project i requires a (so-called "effective") work time of D_{ir} in competency r ($r = 1, \ldots, R$), where D_{ir} is a random variable with a (known) distribution the parameters of which can be estimated in advance. We call that part of a project i that requires a particular competency r the *work package* with index (i, r).

A fixed team of employees $j = 1, \ldots, J$ is assumed as available. For each employee, we suppose that her/his efficiency in each competency r can be quantified as a value γ_{jr}, measuring the fraction of effective work in competency r that s/he is able to deliver within given time, compared to the work of an employee with a "perfect" ability in the considered competency r. If employee j with efficiency γ_{jr} works for y time units in work package (i, r), s/he reduces the effective work time required for work package (i, r) by the amount $\gamma_{jr} \cdot y$. To distinguish y from the effective work time $\gamma_{jr} \cdot y$, we call y the *real* work time.

On the lower decision level, after a decision on a project portfolio has been made, the workload corresponding to the single competencies has to be assigned to the staff over time. Since work times are uncertain, a dynamic policy is required for this part of the planning process. We will consider a specific, fixed policy of this type. It is allowed that several employees contribute to the same work package (i, r), provided that the efficiency value of each of these employees in competency r does not lie below some pre-defined minimum value γ_{min}. Each contribution is weighted by the efficiency value of the assigned employee.

As the objective function, we take a weighted average with fixed weights β_1, β_2 resp. β_3 of three terms: (1) Economic benefit (return etc.), which can be estimated for each project i by a value w_i, its overall value resulting as the sum of the values w_i of the selected projects i. (2) Expected strategic benefit, where strategic benefit measures the degree to which the company engages in future-oriented areas. To quantify this objective, desirability values v_r are assigned to each competency r. The strategic benefit is defined as the weighted sum of the real work times $Y_r(x)$ the overall staff spends in each competency r (given portfolio x), weighted by the values v_r. It is important to note that we take here real work times instead of effective work times; the reason for this choice is that in models on organizational learning (cf., e.g., [3]), competence increment is rather related to real than to effective work. As a consequence, the second objective function is not simply determined by the set of selected projects, but influenced by the staff assignment decision. Observe that the quantities $Y_r(x)$ depend also on the random variables D_{ir} and are therefore random variables themselves. (3) The expected value of the total tardiness $\Psi(x)$ of the selected projects with respect to their due date (given portfolio x). The value $\Psi(x)$ is the sum of the tardiness values $\Psi_i(x)$ of the selected projects i, where, with $C_i(x)$ denoting the completion time of project i under portfolio decision x, the tardiness $\Psi_i(x)$ is defined as $(C_i(x) - \delta_i)^+$ if $x_i = 1$, and zero otherwise. Objective (3) enters with negative sign into the overall objective function, since it is to be minimized.

In total, this produces the objective function

$$\mathrm{E}\left(\beta_1 \sum_{i=1}^{n} w_i x_i + \beta_2 \sum_{r=1}^{R} v_r Y_r(x) - \beta_3 \sum_{i=1}^{n} \Psi_i(x)\right) \to \max,$$

where the first term is in fact deterministic. The constraint is defined by the (given) capacity limits of the employees.

4.2 Solution Technique

On the upper decision level of project portfolio selection, the problem presented in the previous subsection is an SCO problem, which makes the proposed algorithm S-VNS applicable. The feasible set S is in this case the set $\{0,1\}^n$ of possible project portfolios. What remains to be specified is the way the two other aspects of the problem, namely project scheduling and staff assignment, are handled on the lower decision level.

Evidently, our problem formulation contains a (particular) *stochastic scheduling problem* as a special case. Methods for solving stochastic scheduling problems have found considerable interest in the literature; we refer, e.g., to Möhring and Stork [14]. As outlined in [14], a suitable approach to the solution of a stochastic scheduling problem consists in determining a *policy* which acts at so-called *decision points* of the time axis. Roughly speaking, a decision point is a time point where “something happens”. In our case, decision points are $t = 0$ (project start), the ready times ρ_i of the projects, and the (random) completion times of the work packages (i, r). It is important to ensure that the decision prescribed by some policy for decision point t only uses information that is already available at time t; for example, the knowledge of the random work time D_{ir} required by a work package (i, r) that is not yet completed at time t cannot be used in the decision. However, since we assume that the *distributions* of these random variables are known in advance, quantities as $\mathrm{E}(D_{ir})$ may be used by the policy at any time.

```
Procedure Stochastic Scheduling-and-Staffing
set $\tau = 0$; // decision point
repeat until all work packages are completed {
  for all projects $i$ in ascending order of their due dates $\delta_i$ {
    for all yet uncompleted work packages $(i, r)$ with $\tau \geq \rho_i$ in descending order
      of $\mathrm{E}(D_{ir})$ {
      for all employees $j$ in descending order of $\gamma_{jr}$ {
        if (employees $j$ is not yet assigned and $\gamma_{jr} \geq \gamma_{min}$)
          assign employee $j$ to work package $(i, r)$;
  } } }
  determine the earliest time $\tau'$ at which either one of the currently scheduled
    work packages becomes completed, or a ready time occurs;
  subtract, for each work package, the effective work done between time $\tau$ and
    time $\tau'$ from the remaining effective work time (work times by employees
    have to be weighted by their efficiencies);
  set $\tau = \tau'$, and set all assignments of employees to work packages back;
}
```

Fig. 2. Dynamic policy for the lower decision level.

For scheduling the remaining work in the selected projects and to re-assign staff in a decision point, we have chosen a conceptually simple heuristic policy which can be classified as *priority-based* in the terminology of [14]: First of all, projects with an earlier due date have a higher priority of being scheduled immediately than projects with a later due date. Secondly, within a project i, work packages (i, r) with a higher value of $\mathrm{E}(D_{ir})$ (i.e., those that can be expected to consume a higher amount of resources) obtain higher priorities. Finally, for each work package (i, r), employees j with free capacities *and* efficiency value $\gamma_{jr} \geq \gamma_{min}$ are assigned according to priorities defined by their efficiency value γ_{jr} (higher priority for higher efficiency). We present the applied stochastic scheduling-and-staffing algorithm in pseudo-code form in Fig. 2. The chosen policy effects that at each decision point t, the set of employees is partitioned into teams where each team is either assigned to a specific work package, or is (in the case at most one team) "idle" at the moment.

4.3 A Numerical Example

At the moment, we are performing tests using benchmark data on candidate projects, employees, work times, efficiencies and objectives from an application provided by the E-Commerce Competence Center Austria ("EC3"), a public-private R & D organization. The following illustration example is a simplified version derived from this real-world application. Experiments were performed on a PC Pentium 2.4 GHz with program code implemented in MATLAB V 6.1.

The reduced example application (which does not use the EC3 data directly, but is similar in flavor) considers $n = 12$ projects, $J = 5$ employees and $R = 3$ competencies. Ready times and due dates span over a time horizon of 32 months. For modelling the work times D_{ir}, triangular distributions have been used for the first tests, where the parameter estimations map the typical high skewness of work times in R & D projects, the most frequent value lying much closer to the minimum value than to the maximum value. (We do not claim that triangular distribution satisfy the conditions of Theorem 3.1, but they yield meaningful test cases for the algorithm S-VNS nevertheless.) Efficiency values have been determined by a special procedure the description of which is outside the scope of this paper; in the case of the five selected employees for the illustration example presented here, the efficiency values are such that one of the five employees is highly specialized, whereas the efficiencies of the other four employees are more evenly distributed over the three considered competencies. A minimum efficiency score of $\gamma_{min} = 0.25$ has been defined. The economic benefits of the 12 projects have roughly been estimated by the vector $(20, 1, 4, 5, 4, 4, 4, 4, 3, 1, 6, 8)$; for the strategic benefits, the three considered competencies have been weighted by the desirability values 2, 1 and 5, respectively. The time unit is a month. First, we considered the case where the weights for the three parts of the objective function (corresponding to economic benefit, strategic benefit and tardiness) are estimated by the numbers $\beta_1 = 20$, $\beta_2 = 1$ and $\beta_3 = 100$, respectively. This means that work experience of 20 months gained in a competency with desirability 1 is considered as of the same worth as an additional unit of economic benefit, and the loss caused by a delay of one month is considered as equivalent to the loss of five units of economic benefit.

It is not easy to evaluate the results delivered by the proposed algorithm S-VNS, since exact solutions[2] are very hard to determine already for this comparably small problem size: Note that the solutions space for the portfolio decision consists of $2^{12} = 4096$ portfolios, and from the observed variances and the typical differences between objective function values, we estimate that at least 10^4 simulation runs are necessary to compute a sufficiently accurate approximation to the true objective function value $f(x)$ enabling an identification of the optimal solution with some reliability. Of course, even this time-consuming brute-force procedure would not give a guarantee that the true optimum is found. In order to have a yardstick for our heuristic results, we adopted the following pragmatic procedure instead: First, a heuristic solution $\tilde{x}$ was determined by S-VNS, and using the observed SAE variance, an aspiration level c_a was determined in such way that an SAE with sample size $s^{(1)}$ would be above c_a for each solution that is at least as good as $\tilde{x}$, except with a very small failure probability $< 10^{-6}$. Then, we performed complete enumeration of all portfolios $x \in S$ by an SAE with

[2] By the term "exact solution", we refer in the sequel only to the portfolio selection decision, considering the described scheduling-and-staffing policy as fixed. The question whether this policy can be improved will not be treated here.

sample size $s^{(1)}$. Whenever the resulting objective function estimate exceeded the aspiration level c_a for a solution x, we re-evaluated this solution based on a second SAE with large sample size $s^{(2)} \gg s^{(1)}$. For $s^{(1)}$ and $s^{(2)}$, we chose the values 80 and 20 0000, respectively.

By this technique, we found the presumably best solution within a runtime of about 5 hours and 20 minutes. For the weights indicated above, this was the solution $x^* = (0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1)$. We performed now runs of S-VNS where the stopping criterion was a pre-defined maximum number m_{max} of rounds chosen in such a way that each run required about 1 minute computation time only. For each parameter combination, the average relative deviation of the objective function values of the delivered solutions in ten independent runs from that of the (presumably) best solution was computed. We observed that for several parameter combinations, the optimal solution was found in a majority of runs. Let us look at two special parameter combinations. Therein, c_0 denotes the factor of $\sqrt{m}$ in Remark 2 after Theorem 3.1, and Δ_r gives the average relative deviation of the solution quality from the optimum in percent. It can be seen that the relative derivations are rather low:

(i) $s_0 = 5$, $c_0 = 5$, $k_{max} = 12$, $\rho_{max} = 10$, $m_{max} = 10$: $\Delta_r = 1.13\%$.
(ii) $s_0 = 3$, $c_0 = 10$, $k_{max} = 12$, $\rho_{max} = 10$, $m_{max} = 10$: $\Delta_r = 0.86\%$.

Reducing the weight β_3 for objective function 3 from 100 to 20 describes the situation where violations of due dates have less serious consequences. The optimal solution for this weight combination is $x^* = (1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0)$. It is seen that here, the decision maker can risk to include the larger (and more profitable) project no. 1 into the portfolio. Using parameter combination (i) from above, all of ten independent runs of S-VNS provided this optimal solution.

5 Conclusions

We have presented a simulation-optimization algorithm S-VNS for stochastic combinatorial optimization problems, based on the Variable Neighborhood Search metaheuristic, and shown that the solutions produced by S-VNS converge under rather weak conditions to optimality. As an application case, a stochastic project portfolio selection problem involving project scheduling and staff assignment decisions has been described, and the application of the proposed algorithm to this problem has been outlined.

First experimental observations have been obtained, which are of course not yet sufficient for an experimental evaluation of the approach, but show promising results. Tests on a large number of randomly generated instances as well as on real-life instances will be necessary to judge the solution quality achieved by the method. In particular, these tests should include comparisons with other simulation-optimization techniques and verify the found results by statistical significance tests. Within our mentioned application context, we are performing extensive investigations of this type, and a long version of the present paper is planned providing results of this type.

Acknowledgment. Financial support from the Austrian Science Fund (FWF) by grant #L264-N13 is gratefully acknowledged.

References

1. Alrefaei, M.H., Andradottir, S., A Simulated Annealing algorithm with constant temperature for discrete stochastic optimization, *Management Sci.* 45, pp. 748 - 764 (1999).
2. Alrefaei, M.H., Andradottir, S., "A modification of the stochastic ruler method for discrete stochastic optimization", *European J. of Operational Research* 133, pp. 160-182 (2001).
3. Chen, A.N.K., Edgington, T.M., "Assessing value in organizational knowledge creation: considerations for knowledge workers", *MIS Quaterly* 29, pp. 279–309 (2005).
4. Fitzpatrick, J.M., Grefenstette, J.J., "Genetic algorithms in noisy environments", *Machine Learning* 3, pp. 101–120 (1988).
5. Fu, M.C., "Optimization for simulation: theory vs. practice", *INFORMS J. on Computing* 14, pp. 192-215 (2002).
6. Gabriel, S.A., Kumar, S., Ordonez, J., Nasserian, A., "A multiobjective optimization model for project selection with probabilistic considerations", *Socio-Economic Planning Sciences* 40, pp. 297–313 (2006).
7. Gelfand, S.B., Mitter, S.K., Simulated Annealing with noisy or imprecise measurements, J. Optim. Theory Appl. 69, pp. 49 - 62 (1989).
8. Gutjahr, W.J., Pflug, G., "Simulated annealing for noisy cost functions", *J. of Global Optimization*, 8 (1996), pp. 1–13.
9. Gutjahr, W.J., "A converging ACO algorithm for stochastic combinatorial optimization", *Proc. SAGA 2003* (Stochastic Algorithms: Foundations and Applications), Springer LNCS 2827, pp. 10-25 (2003).
10. Gutjahr, W.J., "S-ACO: An ant-based approach to combinatorial optimization under uncertainty", *Proc. ANTS 2004* (4th International Workshop on Ant Colony Optimization and Swarm Intelligence), Springer LNCS 3172, pp. 238-249 (2004).
11. Gutjahr, W.J., Katzensteiner, S., Reiter, P., Stummer, Ch., Denk, M., "Competence-Driven Project Portfolio Selection, Scheduling and Staff Assignment", Technical Report, University of Vienna, Dept. of Statistics and Decision Support Systems (2007).
12. Hansen, P., Mladenović, N., "Variable neighborhood search: Principles and applications", *European J. of Operational Research* 130, pp. 449–467 (2001).
13. Homem-de-Mello, T., "Variable-sample methods for stochastic optimization", *ACM Trans. on Modeling and Computer Simulation* 13, pp. 108–133 (2003).
14. Möhring, R.H., Stork, F., "Linear preselective policies for stochastic project scheduling", Mathematical Methods of Operations Research 52, pp. 501–515 (2000).
15. Nozic, L.K., Turnquist, M.A., Xu, N., "Managing portfolios of projects under uncertainty", *Annals of Operations Research* 132, pp. 243–256 (2004).
16. Norkin, V.I., Ermoliev, Y.M., Ruszczynski, A., "On optimal allocation of indivisibles under uncertainty", *Operations Research* 46 (1998), pp. 381–395.
17. Shi, L., Olafsson, S., "Nested partitions method for global optimization", *Operations Research 48*, pp. 390–407 (2000).

APPENDIX

For the proof of Theorem 3.1, we first show the following auxiliary result:

Lemma 3.2. Suppose that the assumptions of Theorem 3.1 are satisfied. Let $x^* \in S^*$, and consider a fixed cycle starting with current solution $x \in S$ (and $k = 1$ by definition of a cycle).

(a) If $x \neq x^*$, then with P-probability larger or equal to a constant $c > 0$, (i) the medium repeat loop is executed at least until k has that value k^* for which $x^* \in \mathcal{N}_{k^*}(x)$, (ii) the shaking at the beginning of the round with $k = k^*$ yields the solution $x' = x^*$, (iii) in this round, local search (starting with $x' = x^*$) ends with $x'' = x^*$, and (iv) in the "move-or-not" part of this round, $x'' = x^*$ is judged as better than x, such that x^* is exposed to the tournament.

(b) If $x = x^*$, then with P-probability larger or equal to a constant $\bar{c} > 0$, in the first round of the cycle, $x = x^*$ wins against x'', such that x remains equal to x^*, and x^* is exposed to the tournament.

Proof. Let A_1 to A_4 denote the events mentioned under (i) – (iv) in part (a) of the lemma, respectively. We have

$$P(A_1 \cap \ldots \cap A_4) = P(A_1) \cdot P(A_2 \,|\, A_1) \cdot P(A_3 \,|\, A_1 \cap A_2) \cdot P(A_4 \,|\, A_1 \cap A_2 \cap A_3).$$

For proving (a), it suffices to show that all four factors of the product on the r.h.s. have constants > 0 as lower bounds.

Let $k^* = k^*(x)$ be that $k \in \{1, \ldots, k_{max}\}$ for which $x^* \in \mathcal{N}_{k^*}(x)$ holds. (k^* exists because of assumption (a) of Theorem 3.1.) Let $\Delta = \max_{x \in S} f(x) - \min_{x \in S} f(x)$ denote the maximum distance between the objective function values of two solutions in S. Because of assumption (c) of Theorem 3.1 and the fact that the sample size s_0 is constant, there is a probability larger or equal to some constant c_1 that in the test of x against x'' in "move or not", the value of $f(x)$ is underestimated by at least $\Delta/2$, and the value of x'' is over-estimated by at least $\Delta/2$, with the consequence that x appears as the better solution and the "else" branch of "move or not" is executed (i.e., k is increased). With a P-probability of at least $c_1^{k^*-1} \geq c_1^{k_{max}-1} > 0$, this happens in $k^* - 1$ successive rounds. Hence $P(A_1) \geq c_1^{k_{max}-1}$ is shown.

It is immediately seen that

$$\begin{aligned} P(A_2 \,|\, A_1) \;&=\; P(x^* \text{ is selected} \,|\, \text{shaking selects from } \mathcal{N}_{k^*}(x)) \\ &\geq \min_{1 \leq k \leq k_{max}} \; \min_{x' \in \mathcal{N}_k} P(x' \text{ is selected} \,|\, \text{shaking selects from } \mathcal{N}_{k^*}(x)) \end{aligned}$$

and the last is a constant > 0 by the basic property of the shaking mechanism and the finiteness of S.

Concerning $P(A_3 \,|\, A_1 \cap A_2)$, observe that if already during the first execution of the innermost repeat loop, the value of f for the solution $\bar{x} = x' = x^*$ is underestimated and the values of f for all solutions in $\mathcal{N}_1(x^*)$ are overestimated, x^* is judged as a local optimum and becomes therefore the new x''. By assumption (c) of Theorem 3.1, both the probability of underestimating and that of overestimating a function value by the SAE is larger than some constant $c_2 > 0$, hence the mentioned probability is at least $c_2^{\mathcal{N}_1(x^*)+1} \geq c_2^{|S|} > 0$.

Finally, we need a lower bound for $P(A_4 \mid A_1 \cap A_2 \cap A_3)$. If in the test between x and x'' in "move-or-not", the function value of $x'' = x^*$ is underestimated and that of x is overestimated, then x^* wins the test and gets exposed to the tournament. The probability for the mentioned event is at least $c_2^2 > 0$.

Together, this proves assertion (a) of the lemma. For showing (b), let us assume $x = x^*$. Let x'' be the solution yielded by the local search in the first round of the cycle. As in the last paragraph, it follows then that x^* wins against x'' with a probability of at least $c_2^2 > 0$. $\square$

Proof of Theorem 3.1. The core of the proof is similar to the proof of Theorem 4.1 in [13] which refers to a random search algorithm. Within our context, we have to take account of the more complicated situation induced by the VNS–type search. First of all, we show that for arbitrary fixed $x^* \in S^*$, it holds with P-probability one that $x(m) = x^*$ for infinitely many m. Indeed, by applying Lemma 3.2 above to the cycles following the cycle in which an arbitrary fixed round m_0 is contained, we see that in each of these cycles, with P-probability of at least $\min(c, c') > 0$, the solution x^* gets exposed to at least one tournament. Because of the independence of the scenarios by condition (b) of the theorem, it follows that for P-almost all (ξ, η), there exists a round $m_1 > m_0$ with $x(m_1) = x^*$. Hence, with the exception of a set $\Xi^e \subseteq \Xi$ of combinations (ξ, η) with P-measure zero, $x(m) = x^*$ for infinitely many m.

Now let $\tilde{f}_m(x)$ denote the SAE obtained for solution x if x is exposed to the sample ω^m. According to Lemma 3.1, for each x, it holds that $\tilde{f}_m(x) \to f(x)$ for all $\omega \in \Omega$ with the exception of a set Ω_x^e with $\tilde{P}$-measure zero. In other words, for arbitrary fixed $\delta > 0$, $|\tilde{f}_m(x) - f(x)| \geq \delta$ can hold for only finitely many m except if $\omega \in \Omega_x^e$. Therefore, for $\omega \in \Omega \setminus \Omega_x^e$, there is an integer $K_x = K_x(\delta, \omega)$ such that

$$|\tilde{f}_m(x) - f(x)| < \delta \quad \text{for all } m \geq K_x. \tag{1}$$

Let $K = \max_{x \in S} K_x$. Because of the finiteness of S, also $\Omega^e = \bigcup_{x \in S} \Omega_x^e$ has $\tilde{P}$-measure zero. For $\omega \in \Omega \setminus \Omega^e$, inequality (1) implies $|\tilde{f}_m(x) - f(x)| < \delta$ for all $m \geq K$, independently of x. We set now

$$\delta = (1/2) \cdot \min_{x', y' \in S,\, x' \neq y'} |f(x') - f(y')|$$

and assume $x \in S$ and $y \in S$ to be arbitrary solutions with $f(x) < f(y)$. For all $\omega \in \Omega \setminus \Omega^e$, if $m \geq K$,

$$\begin{aligned} \tilde{f}_m(x) - \tilde{f}_m(y) &= (\tilde{f}_m(x) - f(x)) + (f(x) - f(y)) + (f(y) - \tilde{f}_m(y)) \\ &\leq |\tilde{f}_m(x) - f(x)| + |\tilde{f}_m(y) - f(y)| - (f(y) - f(x)) \end{aligned}$$

$$< \ 2\delta - (f(y) - f(x)) = \min_{x', y' \in S,\, x' \neq y'} |f(x') - f(y')| - (f(y) - f(x)) \ \leq \ 0. \quad (2)$$

According to the assertion derived in the first paragraph of the proof, there exists an integer $m_1 > K$ such that in round m_1, a solution $x^* \in S^*$ is exposed to the tournament, i.e., $x(m_1) = x^*$, except for combinations $(\xi, \eta) \in \varXi^e$ with P-measure zero. (Of course, since K depends on ω, the round index m_1 depends not only on (ξ, η), but on ω as well; however, this has no influence on the exception set $\varXi^e$.) In this case, x^* wins the tournament in round m_1 against an arbitrary solution $\hat{x} \notin S^*$, because, by (2),

$$\tilde{f}_{m_1}(x^*) - \tilde{f}_{m_1}(\hat{x}) < 0.$$

Also in the case of $\hat{x} \in S^*$, a solution contained in S^* wins the tournament. Hence, in any case, $\hat{x}_{m_1+1} \in S^*$. Since this situation also prevails in all subsequent rounds, we obtain $x_m \in S^*$ for all $m \geq m_1$. □