Adaptation in Dynamic Environments: A Case Study in Mission Planning

[†]Lam Thu BUI, [‡]Zbigniew MICHALEWICZ, [‡]Eddy PARKINSON and [‡]Manuel Blanco ABELLO

***Abstract*— Many random events usually are associated with executions of operational plans at various companies and organizations. For example, some tasks might be delayed and/or executed earlier. Some operational constraints can be introduced due to new regulations or business rules. In some cases, there might be a shift in the relative importance of objectives associated with these plans. All these potential modifications create a huge pressure on planning staff for generating plans that can adapt quickly to changes in environment during execution. In this paper we address adaptation in dynamic environments.**

Many researchers in evolutionary community addressed the problem of optimization in dynamic environments. Through an overview on applying evolutionary algorithms for solving dynamic optimization problems, we classify the work into two main categories: (1) finding/tracking optima and (2) adaptation and we discuss their relevance for solving planning problems. Based on this discussion, we propose a computational approach to adaptation within the context of planning. This approach models the dynamic planning problem as a multi-objective optimization problem and an evolutionary mechanism is incorporated, this adapts the current solution to new situations when a change occurs.

As the multi-objective model is used, the proposed approach produces a set of non-dominated solutions after each planning cycle. This set of solutions can be perceived as an information-rich data set which can be used to support the adaptation process against the effect of changes. The main question is how to exploit this set efficiently? In this paper we propose a method based on the concept of centroids over a number of changing-time steps, at each step we obtain a set of non-dominated solutions.

We carried out a case study on this proposed approach. Mission planning was used for our experiments and experimental analysis. We selected mission planning as our test environment because battlefields are always highly dynamic and uncertain and can be conveniently used to demonstrate different types of changes, especially time-varying constraints. The obtained results support the significance of our centroid-based approach.

***Index Terms*— adaptation, dynamic environments, evolutionary algorithms, multi-objective optimization**

I. INTRODUCTION

Real-world problems often contain many uncertain and dynamic factors; i.e., air traffic scheduling is usually affected by unexpected events such as bad weather or emergencies, military mission planning might have to endure delays or failures of capabilities. Therefore, it is unlikely that any solution found for these problems would stay valid for a long time horizon. These uncertain and dynamic factors require an adaptive mechanism to introduce changes to the current solution.

Evolutionary algorithms (EAs) have been a popular means for solving optimization problems in dynamic environments [27]. To date, there have been a significant number of techniques proposed within the framework of EAs to solve Dynamic Optimization Problems (DOPs). These techniques include diversity-based approaches, either explicit, such as random immigrant and hyper-mutation, or implicit, such as multi-objectization, memory-based, or multi-population based. The readers are referred to [10] for a more detailed survey in this area.

It is interesting to see that the past work on EAs for dynamic optimization problems can be divided into two broad categories: (1) finding/tracking optima over time and (2) adapting against the effect of changes. The approaches from the first category deal with each change in the environment by generating a new population of solutions and evolving them further to find and track the new optima. Meanwhile, the approaches from the second category address different circumstances — for example, where a part of the solution has already been executed by the time the change occurs. In these approaches it is necessary to address the issue of adaptation from the current solution in which any derivation from the current solution has to take into account the associated costs. While the approaches of the first category are more related to the issue of theoretical convergence, the approaches of the second one are more practical and have huge applicability to many real-world problems [34].

Note also, that given a change in the environment where a solution is already in execution, re-planning of the whole solution might not be feasible or might incur a high cost. So the adaptation process should ensure meeting the primary objectives of the problem while keeping the cost of adjusting the solution to a minimal level. In other words, the existence of multi-objectivity within this adaption process is natural. Given the importance of this issue, however there seems a lack of dedicated research to develop computational frameworks for dealing with this issue.

Further, there are three types of changes which have been considered in the literature. These are *time-varying objective functions*, *time-varying parameters*, and *time-varying constraints*, see section II for detailed descriptions. While the first and the second types have been quite popular when using EAs

Manuscript received xxxx, 2010; revised xx-xx-2010. This work was supported by Australian Research Council via the ARC Discovery Grant No. DP.

email: lam.bui07@gmail.com, zbyszek@cs.adelaide.edu.au, edward.parkinson@adelaide.edu.au, manuel.abello@adelaide.edu.au

[†] Department of Software Engineering, Le Quy Don University, 100 Hoang Quoc Viet Street, Cau Giay, Hanoi, Vietnam. [‡]School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia; also at the Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland, and at the Polish-Japanese School of Information Technology, ul. Koszykowa 68, 01-222 Warsaw, Poland

to deal with these changes, the third type of time-varying (or dynamic) constraints has received much less attention. This is a bit surprising as this third type represents a typical scenario in a business or industry environment. Note that appearance on a new constraint (or a modification of an existing one) may result in infeasibility of the current solution; a removal of a constraint may result in the appearance of a better solution. So in this paper we will also address this type of changes.

As indicated earlier, we investigated the issue of adaptation within the scope of scheduling and planning, especially the problem of Resource Constraint Project Scheduling — RCPS, since this problem domain has been a popular one for studying adaptation in dynamic environments [37]. We reformulated the RCPS to reflect some dynamic factors and (in the context of a planning problem) called it Adaptive Planning Problem (APP). For this problem, there is a need to prepare adaptive plans to deal with changes that might happen during execution. The question is that given the current (partially executed) plan, how to generate new plans that can satisfy both objectives: keeping the execution within its time-limit (original objective) while maintaining minimal cost of alteration (additional objective)?

In our approach we adapt the current plan in a reactive-style using an evolutionary algorithm. For any component which has already executed or is in progress, it will not be adjusted or rescheduled. In this way the rescheduling process becomes simpler over time since the number of tasks to be scheduled decreases. To assist in the decision making process, we use the second objective as an additional indication for selection of a new plan. A set of plans is obtained by analysing trade-off between time and cost of re-allocated resources. Further, since changes might happen frequently during execution of a plan, it should be worthwhile to use the results obtained in the past to initialize the population in order to speed up the adaptation process. The past information is contained in a large number of non-dominated solutions. In this paper we propose a novel mechanism where for each set of non-dominated solutions obtained before a change, we calculate its centroid which shows the overall tendency of the whole set. The set of centroids will be used as an initialization factor for the new population, whenever a change happens. We call it the Centroid-Based Adaptation (CBA).

The selected case study was given within the context of military mission planning, since it can be easily formulated as an instance of the APP. Military missions are usually uncertain and dynamic. The main objective is to minimize the execution time of the mission with a limit on available capabilities. Two objectives are proposed: the execution time of the plan, and the cost of operating capabilities. Also, several time-varying constraints are proposed, including execution time, the failure of capabilities, and change of task-relationship network. The performance of the proposed approach was analyzed and discussed. We validated CBA against three other EA methods: randomly initializing, using the last population, and non-dominated solutions only from the last population. The obtained results strongly support our proposal with a consistent and better performance on all three types of changes.

The paper is organized as follows: an overview of evolutionary algorithms in dynamic environments, evolutionary multi-objective optimization, and project scheduling are presented in sections II, III, and IV, respectively. The problem formulation and the proposed method are introduced in section V. A case study on how can we exploit trade-off solution to facilitate the adaptation process in section VI. The last section is devoted to the conclusion of the work and some lessons learnt.

II. DYNAMIC OPTIMIZATION PROBLEMS AND EVOLUTIONARY ALGORITHMS

A characteristic of dynamic optimization problems (DOPs) is change over time. Under the effect of changes, the search space is likely to be altered, and the current solution might be no longer optimal, or may become infeasible. As discussed briefly in the previous section of the paper, the paper defines the following categories:

- **Time-varying objective functions**: For example enemy units arrive at a location, making some parts of the objective more difficult. The objective function is not constant over time. Therefore, the objective value of a solution can be different at different times. This usually causes the occurrence of new optima. This category of change has been popular for research in EAs for tracking optima over time, i.e., the moving peak benchmark (MPB) problem [10].
- **Time-varying variables**: An example problem of this category is dynamic machine scheduling where unexpected new jobs arrive. A time-varying variable can be used for determination of the objective value, but can be a late addition or change. To date, dynamic job-shop scheduling has been a popular test case in the literature [37].
- **Time-varying constraints**: For example the precedence relationship of tasks. The objective function and variables do not change for this category, however, the constraints may change over time. This category of change does not change the fitness landscape driven by the objective function, but it will affect the areas of feasibility. It is interesting to note that this category of change has not been analyzed in detail in comparison to the other two categories.

As indicated earlier, there has been an increasing number of works applying EAs to solve optimization problems in dynamic environments. This is mainly because of the way EAs mimic natural evolution, a number of individual solutions are allowed to compete and evolved over time. At the time of a bounded change, this population of solutions (now at fairly different areas of the search space) can easily evolve towards the new optima. In general, the EA approaches for DOPs are categorized into two broad areas: finding/tracking optima and adaptation. We discuss these two areas in the following subsections.

A. Finding/tracking optima

This research direction focuses on the convergence aspect that allows EAs to quickly pinpoint the moving optima and track them over time. It usually uses the last population before the problems change as a starting point for the new search space. However, for this direction, it should be noted that the change should be bounded, so that the effect is not radical and the previous information in the individuals of the previous population is still useful. The first and second categories of change are usually more suitable for this. To address the convergence issue, four popular approaches are considered:

- **Generating diversity after a change**. Diversity is considered as an important element for EAs to effectively track optima after a change. The most natural way to do this is to *reinitialize* the population after a change. However, with some bounded changes, the new search space is not radically different from the previous one. In such cases, reinitialization might slow down the convergence process; hence reusing the previous population might be a better option. Note that the last population might be driven towards the area surrounding the previous optima. However, with a change, the new optima might be at a different location. Therefore, if we use the last population, it needs some diversity. A popular approach is the use of *hyper-mutation* [16] where the mutation rate is set to a high value for a limited number of generations, or just the first generation and then decreased over time. Note that a very high mutation rate may result in a reinitialization of the population, whereas too low mutation rate does not help to boost diversity of the population.
- **Maintaining diversity during the run**. Diversity is maintained over time to make sure the population is diverse enough to recover from the effect of change. Typical approaches include *random immigrant* [23] where new individuals are inserted into the population on regular basis, *multiobjective-based method* utilizing diversity as the second objective [14], and the *thermodynamic genetic algorithm* where an entropy-based value for niching is used during the optimization process [35].
- **Memory-based approaches**. A memory is used to record information of the past optima. This information is useful if future optima return to the vicinity of previous optima. This can be done either in explicit or implicit styles. For the explicit one, whenever a change happens, some individuals from memory will be inserted into the population [39]. Meanwhile, for implicit approaches, redundant representations, such as diploidy, are used to generate solutions [22].
- **Multi-population approaches**. Several sub-populations are evolved together to cover several promising areas of the landscape. In this way, it is hoped that the whole system will be able to quickly find new optima when experiencing a change. Also, with this approach, several optima can be tracked simultaneously. Some typical examples are: self-organizing scouts [11]; the multi-national approach [43]; and the shifting balance approach [48].

B. Adaptation

The task of finding and tracking optima does not take into account the situation where a solution is already in use. Some components of the solutions are occupied and can not be altered or there is a high cost associated with such alteration. Hence, it is essential to adapt the remaining parts of the solution against the change.

Generally speaking, adaptation is a process of adjusting to the new conditions to keep the system functioning properly. Since the world is dynamic, adaptation is a vital process for many existing systems from biological, ecological to social organizations [40]. After a change in the weather, a species might take several generations to develop new abilities to deal with new weather conditions. A company might need to change some components of their business solution to deal with new market conditions. Although adaptation is a complex process and different from system to system, it needs to be executed in an incremental style in which the current in-use components can not be removed without careful consideration of cost.

In this area, scheduling and planning has emerged as a popular test problem in which dynamic factors are common, such as arrival of new jobs (or tasks), disruption of machines (or resources), time delays in executing jobs. There has been a considerable number of works on this topic [25], [45], [53], [37]. In general there are three classes of method for tackling this adaptation issue:

- **Reactive**: In these methods a pre-optimized solution is used as a baseline for scheduling. Whenever a change happens, this baseline solution is revised or repaired to adapt to new conditions. However, this baseline is obtained without any anticipation about the uncertainties. The repairing or revising process is usually repetitive and is considered as a local search process [31], [45]. Since this approach is based on existence of the baseline solution, some researchers have labeled it as 'predictive-reactive'.
- **Proactive**: These methods take into account some assumptions (anticipation) about the uncertainties such as the bound or level of changes, or the probabilistic distribution of changes in order to find the most suitable solution. An example proactive approach is adding slack time to a solution, to make it more robust. The proactive solution is usually obtained via a sensitivity analysis (e.g., using the Monte Carlo simulation). This is also considered as the robustness analysis process [46].
- **No-baseline**: In these methods there is no baseline solution set in advance. A new search process is carried out to find the new solution adapting to new conditions (it is similar to the re-initialization process).

Also, the use of multi-objectivity has been also considered as a tool for adaptation [12], [50], [49]. This is because the consideration is not just given to the original objective, but also to the associated impacts of adjusting the solution. We will return to this point later in the paper. Overview: Scheduling and Resource Allocation

III. Evolutionary Multi-objective Optimization

Real-world problems often have multiple and conflicting objectives e.g., one would like to have a good quality car, but also to spend small amount of money to buy it. A solution to a multi-objective problem (MOP) is called a Pareto optimal solution if "there exists no other feasible solution which would decrease some criterion without causing a simultaneous increase in at least one other criterion" [17].

This definition of Pareto optimality implies the existence of several trade-off solutions (called the Pareto optimal set, or Pareto optimal front (POF) for the plot of the vectors of decision variables corresponding to these solutions in objective space. Multi-objective evolutionary algorithms (MOEAs) are stochastic optimization techniques to find such Pareto optimal solutions for a particular problem [18]. Because, for a particular problem, many Pareto optimal solutions may exist, MOEAs are fundamentally different with respect to single-objective optimization algorithms.

More precisely, in a *k*-objective optimization problem, a vector function $\overrightarrow{f}(\overrightarrow{x})$ of *k* objectives is defined as:

$$\overrightarrow{f}(\overrightarrow{x}) = [f_1(\overrightarrow{x}), f_2(\overrightarrow{x}), ..., f_k(\overrightarrow{x})] \quad (1)$$

where $\overrightarrow{x}$ is a vector of decision variables in the *n*-dimensional space $\mathbb{R}^n$. Each individual (solution) is represented by a vector $\overrightarrow{x}$ and its merit is given by an evaluation vector $\overrightarrow{f}$. Hence the comparison of solutions is not as straightforward as it is in single-objective optimization problems.

Let's introduce an additional concept essential for evaluating individuals in a population: the concept of dominance. An individual $\overrightarrow{x}_1$ dominates $\overrightarrow{x}_2$ if $\overrightarrow{x}_1$ is better than $\overrightarrow{x}_2$ when measured on all objectives. If $\overrightarrow{x}_1$ does not dominate $\overrightarrow{x}_2$ and $\overrightarrow{x}_2$ also does not dominate $\overrightarrow{x}_1$, they are non-dominated. Further, let us introduce the following notation: $\overrightarrow{x}_1 \preceq \overrightarrow{x}_2$ if $\overrightarrow{x}_1$ dominates $\overrightarrow{x}_2$. Also, for scalars *a* and *b*, $a \lhd b$ if *a* is better than *b* (similarly, $a \rhd b$ if *a* is worse than *b*, and $a \ntrianglerighteq b$ if *a* is not worse than *b*). With this notation the domination concept is formally defined as follows.

Definition 1: $x_1 \preceq x_2$ *if the following conditions hold*:
1. $f_j(x_1) \ntrianglerighteq f_j(x_2)$, $\forall\, j \in$ [1, 2, ...,*k*]
2. $\exists\, j \in$ [1, 2, ...,*k*] *in which* $f_j(x_1) \lhd f_j(x_2)$.

In general, if an individual in a population is not dominated by any other individual in the population, it is called a non-dominated individual. All non-dominated individuals in a population form the non-dominated set (as formally described in *definition 2*):

Definition 2: *A set S is said to be the non-dominated set of a population* P *if the following conditions hold:*
1. $S \subseteq P$
2. $\forall\, s \in S, \nexists\, x \in P \mid x \preceq s$

When the set *P* represents the entire search space, the set of non-dominated solutions *S* is called the *global Pareto optimal set*. If *P* represents a sub-space, *S* is called the *local Pareto optimal set*. There is only one global Pareto optimal set, but there could be multiple local ones. In general, we simply refer to the global Pareto optimal set as the Pareto optimal set.

Over the years, many MOEAs have been developed. Usually they are classified into two broad categories: MOEAs with and without elitism. With the elitism approach, MOEAs employ an external set (the archive) to store the non-dominated solutions after each generation. This set is a part of the next generation. With this method, the best individuals in each generation are always preserved, and this way helps the algorithm gets closer to the POF. Algorithms such as SPEA2 [54], PDE [2] and NSGA-II [19] are examples of this category. In contrast, the non elitism approach has no concept of elitism when it does selection of individuals for the next generation from the current population [55]. Examples of this category include VEGA [41] and NSGA [18].

IV. Resource constrained project scheduling problems

In the section we provide an overview of resource constrained project scheduling problems with emphasis on their multi-objective aspects and adaptability.

1) Overview: Scheduling and Resource Allocation (SRA) is a decision-making support process involving assignment and allocation of limited resources to tasks (sometimes referred as jobs, operations or activities) over time under certain constraints. It also needs to deal with defining which tasks to be executed at a specific time [38]. Metaheuristic techniques providing approximate solutions are often used for SRA because many scheduling problems are NP-hard [30]. In these situations the majority of exact methods fail to obtain optimal solutions when the problem size is considerably large [5].

In scheduling and resource allocation problems there are usually several types of resources available (machine, people, money, etc). Traditionally, these problems are considered in a situation where there is a scarcity of resources when scheduling and it is usually referred to as *resource constraint project scheduling problem* — RCPS. For this problem, tasks are characterized by several aspects such as task duration, and required resources. Tasks can be involved in several execution modes and are constrained by a precedence graph. Only one mode is performed at a time. Algorithms for RCPS are expected to find an optimal schedule that minimizes the processing time (called the makespan). In general formulation, this is a NP-hard problem [8]. Further, there are more practical approaches that can be transformed in the form of a RCPS such as production sequencing, timetabling, and flight scheduling.

In general, RCPS can be modeled as mixed integer programming problem. Therefore, conventional linear programming, such as the simplex method, is not really suitable, and many researchers have developed a number of approaches over recent years. Broadly speaking, these approaches can be classified into two categories: exact and approximate methods. For exact methods, the final solution will be the optimal one for the RCPS. Some typical approaches of this category are branch and bound [20], Lagrangean relaxation [21], and dynamic programming [15]. However, exact methods usually face an issue of execution time with large scale problems (the number of tasks should not exceed 60, according to [5]). Therefore, approximation methods are preferred instead, such as a variety of heuristic/meta-heuristic techniques, e.g., priority-based, truncated branch and bound, sampling techniques, local search techniques, tabu search, simulated annealing, scatter search, and evolutionary algorithms [29], [26], [32]. For stochastic project scheduling problems, several approaches have been proposed such as dynamic PERT networks [6], or MDP-based Q-learning [3].

2) Multi-objective RCPS: Further, it has been surprising that although RCPS inherently possesses a feature of multi-objectivity, the literature of RCPS is dominated by work considered only single objectives [42], [47]. There exist a number of possible objectives for RCPS such as time, cost, resource balancing, robustness, etc. These objectives might conflict with each other to different degrees. It is clear that for multi-objective optimization, finding good (optimal) solutions is not the only consideration, but also addressing the trade-off between objectives among obtained solutions.

Perhaps, the common approach in dealing with multi-objective RCPS (also for machine scheduling [36]) is the weighted-sum [1]. For this, objectives are summed according to a predefined vector of weights. However, this approach systematically faces several issues, such as objective scaling, and further it is not easy to address the matter of trade-off analysis. Alternatively, Pareto-based approaches are used to obtained a set of trade-off solutions in a single run [47], [7]. Therefore, these approaches seem to be more suitable for solving multi-objective RCPS.

3) Adaptation in solving RCPS: As indicated in the previous section, multi-objectivity is an inherent characteristic of adaptation. To date, there have been several papers on this. A proactive example is described in [12], the set of trade-off solutions with risk as an additional objective is used; also in this manner, [50] proposed to use the obtained trade-off solutions as alternatives when changes happen. In [49], a reactive adaptation technique was proposed. Two objectives are defined: one is the make-span; while the second is the magnitude of derivation from the baseline schedule, after every change (called reliability in the paper), a set of trade-off solutions is obtained, one will be selected subject to the decision makers preference towards reliability.

It is interesting from these works that they used multi-objectivity purely for proposing alternatives for the selection after the change. There has not been any work, especially for the reactive style, that exploit the characteristics of multi-objectivity to facilitate the search in the adaptation process. One such characteristic is that multi-objective approaches usually offer a set of trade-off solutions. This set of solutions can provide a guide to the tendency of the search over time. The question is how can we exploit this tendency to facilitate the adaptation process? This becomes the focal point of the research reported in this paper.

V. Methodology

We start with formulating a planning problem. Since it is dealing with the matter of adaptation, we call it the *Adaptive Planning Problem* (APP). By adaptation, we mean adjusting the baseline plan, which tackles any infeasibility caused by changes in real-time. Therefore, there are two key aspects that need to be addressed here:

- **In-use situations**: We propose to solve this problem in a reactive manner. Therefore, it is important to take into account the fact that there might be some parts of the plan that are already in use when the change occurs. These can not be alternated or will have a high cost. Further, some resources might have been transferred to the location of the assigned task; any adjustment can result in a cost. Therefore, adaptation from the baseline plan needs to consider the associated cost caused by adjusting in-use resources.
- **Objective functions**: As we have pointed out already, adapting the current plan can not focus on only one objective, such as the plan's execution time. The use of more objectives will give decision makers more choices to make a decision on the results offered by the adaptation process. In the literature of RCPS, some authors proposed to use the derivation from the baseline plan as the second objective. However, in problems such as APP, one unexpected change can cause the whole plan to become infeasible. Therefore, we should define a more concrete objective for this issue and the natural one is the cost of operating capabilities triggered by adjusting the plan. Note that this cost should not be confused with the cost of resources themselves. It can be the cost of moving resources in terms of money, fuel, or human cost.

A. Mathematical formulation of APP

The problem formulation is described as follows:

- **Inputs:**
  - A set V of N tasks: $V = V_1, V_2, ..., V_N$, these are non-pre-emptive. Each V_i will have:
    - A durations d_i
    - A vector rr_i of required resources by tasks: $rr_i = \{rr_{ij}\}$ with $j = 1, ..., M$ (M is the number of resource types)
  - A network G of tasks where nodes and arcs represent the tasks and the precedence relations respectively: $G = (V, E)$. $Pred(j)$ defines a set of direct predecessors, while $Succ(j)$ is the set of direct successors of task j. A dummy node 0 represents the starting point (central base)
  - A vector c of operational costs $c = \{c_{i,j,k}\}$, $i = 0, .., N$; $j = 0, ..., N$, and $k = 1, ..., M$. Here $c_{i,j,k}$ is the cost of moving resource type k from task i to task j. $c_{i,0,k} = 0 \forall i, k$ - no cost is imposed on the return of items to base
  - A set R of M resources $R = \{R_1, R_2, ..., R_M\}$
- **Parameters:**
  - A vector of start time st: $st = \{st_i\}$, with st_i is the starting time of task i and $i = 1, ..., N$
  - For each R_i at time t, a vector of locations for each item of a resource type l_{it} is defined to indicate where the item is located (or the task index). A zero value means the item is at the central base): $l_{it} = \{l_{ijt}\}, j = 0, ..., R_i$
  - For each R_i at time t, a vector of previous locations for each item of a capability type lc_{it} is defined to indicate where the item was from (or the task index). A zero value means the item is at the central base): $lc_{it} = \{lc_{ijt}\}, j = 0, ..., R_i$
  - For each R_i at time t, a vector of locations for each item of a capability type m_{it} is defined to indicate

if the item was moved or not: $m_{it} = \{m_{ijt}\}, j = 0, ..., R_i$

$$m_{ijt} = \begin{cases} 1 & l_{ijt} \neq lc_{ijt} \\ 0 & \text{otherwises} \end{cases} \tag{2}$$

- A vector $r_t = r_{it}, i = 1, ..., M$ presents the current amount of resources being used at time t
- Indices of the tasks: $I = \{I_1, I_2, ..., I_N\}$ (a schedule)

- **Constraints:**
  - Time constraint: If task i is predecessor of task j, then it needs to be completed before starting task j

$$st_i + d_i \leq st_j \tag{3}$$

$\forall j$, and $\forall i \in Prec(j)$

  - Resource constraint: the amount of being-used resources can not exceed the total amount

$$r_{it} \leq R_i \tag{4}$$

$\forall i$ and t

  - Precedence constraint: A task needs to be scheduled before its successors.

$$I_i \notin Succ(I_j) \forall i, j | I_i \leq I_j \tag{5}$$

- **Objective functions:**
  - Makespan (f_1): Minimization of the start time of the last task to be scheduled

$$f_1 = st_{\overline{N}} \tag{6}$$

where $\overline{N}$ is the last task to be scheduled

  - Cost of resource operations (f_2): cost of moving resources between locations of tasks. During the time of executing the schedule (t=1,...T) with T is the maximal number of time steps, if a resource is moved from a location to location, the cost of moving between locations is added to the total cost.

$$f_2 = \sum_{t=1 \to T} \sum_{j=1 \to M} \sum_{k=1 \to R_j} m_{jkt} * c_{lc_{jkt}, l_{jkt}, j} \tag{7}$$

- **Outputs:**
A schedule st based on the obtained index I $st = \{st_1, st_2, ..., st_N\}$
- **Dynamic factors**
  - **Duration**: Dynamic duration of a task V_i is defined as $d'_i(t)$. It is reasonable to consider this change following a probabilistic distribution that usually is $N(d_i, \delta)$, where N is the normal distribution with the mean as the pre-defined duration d_i. Constraint Eq. 3 is rewritten as follows

$$st_i + d'_i(t) <= st_j \tag{8}$$

  - **Availability of resources**: we use a sign function to indicate the availability of resources:

$$l'_{ijt} = sign(t) * l_{ijt} \tag{9}$$

where $sign(t)$ is the sign function returning either +1 or -1, a negative value means unavailable but at the location $|l_{ij}|$.

  - **Precedence network**: The dynamic function representing this change is defined as reversing the relationship between two tasks on network G.
    * A function $rev(i, j)(t)$ is defined for this change defining reverse precedence between two tasks i and j.
    * $i, j \in V'(t)$ where $V'(t)$ is the set of un-executed tasks.
- **Parameters after a change**
Structures of l_{it}, lc_{it}, m_{it} and r_t remain unchanged. The only change is applied to the indices in which $I = I_1, I_2, ..., I_{N'}$ where N' is the number of tasks in $V'(t)$

B. An evolutionary multi-objective approach for APP

The use of an additional objective for APP is to facilitate the adaptation process. Hence, we need to design a multi-objective approach that can offer a set of trade-off solutions for the commanders and their staff to make the decision. Here, we propose to use a GA approach using dominance relations. The algorithm starts with a population being initialized by techniques proposed in the next section. This population will be evolved over time. During the evolution process, all good solutions are preserved.

To perform this task, we employ the non-dominated sorting mechanism as proposed in NSGA-II [19] where the parent population and offspring are combined and sorted in order to generate a population for the next generation. Selection of solutions for producing offspring is also performed as in NSGA-II where a scheme of crowding tournament selection is used. However, the crossover and mutation operations are redesigned since the original operators for NSGA-II are not suitable for our problem.

1) Solution representation: Representation is an important issue to our problem. In our problem, a complete solution must contain information for the schedule of task execution. It contains an indirect representation of a schedule. The indices of this sequence indicate the order of scheduling, while the element at each index is the ID of a task in $V'(t)$ to be executed.

$$S = (I_1, I_2, ..., I_{N'})$$

2) Schedule generation and evaluation: We simply select the Serial Sequence Generation Scheme (SSGS) as the scheme for our schedule generation (Figure 1). For this scheme, when a task is scheduled, it is necessary to check whether it will violate the precedence relation and resource constraint. For the details of SSGS, the readers are referred to [24]

Evaluation of a solution involves calculation of two objectives. The task list is to make the schedule and therefore the makespan. The cost objective will be determined by simulating the plan with regards to the current in-use plan.

3) Genetic operators: The crossover operator is crucial for the behavior of GAs. Two solutions are selected and their features are combined to generate two offspring. Similar mechanisms occur in biology, this operator allows children to inherit characteristics from their parents. However, from

```
Procedure SSGS
Begin
Determine a set of eligible tasks $\varepsilon$
For g=1 to $N'$
Select a task $j \in \varepsilon$
Determine $t = max\{0, max\{st_i + d_i | i \in Pred(j)\}\}$
Schedule j at the earliest precedence - and resource- feasible start time $t' > t$
Set $st_j = t'$ and update $\varepsilon$
End
End
```

Fig. 1. Serial sequence scheduler procedure.

a search and optimization perspective, this operator provides exploitation ability to the algorithm in which offsprings are generated in sub-spaces around parents.

We apply a two-point crossover strategy to our algorithm. This means that for each crossover operation, we need to select two crossing points. Let us assume that we select two parents P_1 and P_2 for crossover and two newly generated offspring O_1 and O_2. There are three parts separated by two crossing points. O_1 takes the first part from P_2 as its first part. In turn, O_1 inherits from P_1 for its second part, and P_2 again for its third part. The process is the same for O_2 with the inverse order: $P_1- > P_2- > P_1$. However, inheriting of the second and third parts of the offspring is different from the first part in which all elements that already exist in the previous parts will be eliminated in order to satisfy the precedence constraint. This is done similarly as the PMX crossover [33].

Also, mutation is another important operator. It randomly changes the values of one or more genes in the chromosomes according to a certain distribution. This bio-inspired operator helps to reintroduce some genetic materials lost during the evolutionary process and some variability to the population. In search/optimization, this operator strengthens the exploration ability of the algorithm. It might help the algorithm to search unexplored areas of the search space. The mutation operation works as follows two consecutive genes are swapped with a predefined probability, if the newly formed sequence of tasks is precedence feasible.

C. Starting a population after a change

The initial population is very important in such time-demanding scenarios as dynamic planning. A good initialization will give the search a quick convergence towards the optimal solution. For APP, a natural way should be to start with random initialization of the initial population as done in [49] for RCPS. This method is very straightforward to implement, but gives a slow convergence during the adaptation process since the population is started from scratch. An opposite view is also taken when adapting the current plan for APP that is to start the population from the last population obtained from the previous change. This helps to speed up the search if the new optima is somewhere close to the area of the old population. However, if the effect of the change is large, the old population becomes entirely infeasible, this method turns to be the random initialization method.

Here, we propose a new method for initializing the population. With the existence of a non-dominated set after each change, there is an opportunity to exploit information from this set to facilitate initialization of the population. The difference from single-objective problems is that here we can use an area of attraction (the area occupied by the set of non-dominated solutions) to initialize the population instead of a single point (the single best solution).

In many single-objective cases, changes over time will make the optima jump among the peaks. Therefore, it is worthwhile to keep track all the optima found in the past. Hence when a change happens, we can start searching from those past optima in order to quickly find the new optimal solution. However, for APP, the challenge is that we are dealing with a set of non-dominated solutions over time. Two issues arise:

- **Memory limitation**: One option might be to keep all solutions, created over time. However, this requires a large amount of memory and will slowdown the whole system. So, this is not really a suitable approach.
- **Selection**: Given enough memory, keeping all non-dominated solutions is possible, but we face a selection problem, since the size of the initial population is limited. Hence we need to decide how to select solutions and that is not a trivial matter.

To ease these issues, we introduce the concept of centroid to store the set of non-dominated solutions. In a broad sense, we just need to know the tendency of the previous population, via their non-dominated solutions, rather than all the solutions themselves. We define the centroid C(t) of a set of non-dominated solutions $\overline{P(t)}$, is defined as an average vector of this set. Each element of the centroid is calculated as follows

$$C_i(t) = \frac{1}{\overline{N}} \sum_{j=1,...,\overline{N}} x_i^j(t) \tag{10}$$

where $x^j(t)$ is a non-dominated solution that belongs to $\overline{P(t)}$ and $i = 1, ..., n$. $\overline{N}$ is the size of $\overline{P(t)}$, n is the dimension of the search space, and $\overline{N}$ is the size of $\overline{P(t)}$.

Note that, this centroid is not necessarily precedence-feasible. We accept this because the purpose of using the centroid is just for determining the tendency of a solution set. If this set of centroids is included to the population, the search process will quickly locate the area of attraction and hence speed up the search. An example is given in Figure 2 to demonstrate this idea. In both left and right graphs, the dots represent the centroids of previous non-dominated sets. The shaded area P represents the previous area of the non-dominated set. There are two scenarios: (1) the new area of the non-dominated set C is similar to the previous area, represented by a centroid, meaning the centroid is feasible. In this case, the search will quickly jump to this area via offspring of the centroid. (2) the new area C (in the right graph) does not overlap with the centroid, but it is quite close. The centroid is now infeasible, but a small mutation can generate solutions that belong to the new area C.

Inclusion of *C* to *P(t)* is done as follows: a set $C'(t-1)$ is adapted from *C(t-1)* as in Eq 11

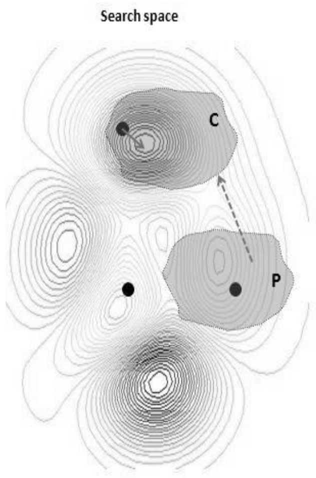
$$C'_i(t-1) = \begin{cases} C_i(t-1) & C_i(t-1) \text{ is precedence-feasible} \\ rnd(C_i(t-1)) & \text{otherwise} \end{cases} \quad (11)$$

where $rnd(C_i(t-1))$ is a feasible node (following constraint 3) that randomly mutated from $C_i(t-1)$

Therefore, $P(t)$ is defined as

$$P(t) = C'(t-1) + P1(t) \quad (12)$$

where $P1(t)$ is the set of $N - |C'|$ solutions randomly chosen from P(t-1). We call our approach Centroid-based Adaptation (CBA).



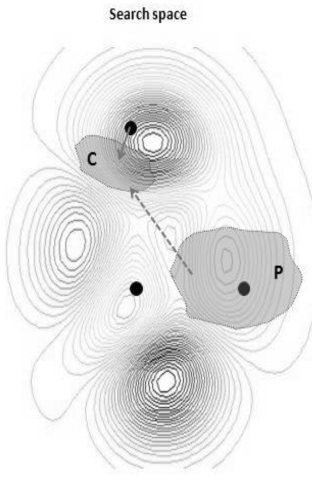


Fig. 2. A Demonstration of Centroid-based Adaptation: the dots represent the centroids of previous non-dominated sets. The shaded area P represents the previous area of the non-dominated set. The shaded area C represents the current area of the non-dominated set

VI. A CASE STUDY

A. Military Mission Planning

1) Overview of planning process: Mission planing is a decision making process in which the commander's intent is materialized. It is a vital element in military command and control aiming at providing a solution that implements the commander's intent, establishes activities, time or conditions for the operation, allocates resources, assigns tasks and coordinates subordinates. This is a complicated process that involves two aspects: (1) Science that deals with measurable factors such as capabilities, techniques and procedures, and it is closely related to the analytic decision making; and (2) Art where the intuition of the commanders about the relationships between friendly forces, enemies and environment as well as the effects of the operation on the solders are the focus and it can be considered as a kind of the intuitive decision making. Mission planning is usually done for a matter of urgency or within a short time frame of a planning horizon [44].

It is quite common in military domain that each level in mission planning corresponds to a level of conflict: *strategic*, *operational*, and *tactical*, although the borders between these three is not always clear. The *strategic level* of a conflict involves determining national or alliance security objectives and developing and using national resources to accomplish those objectives. It establishes strategic military objectives, sequences the objectives, defines limits and assesses risks for the use of military and other instruments of power. Developing strategic plans to achieve the objectives providing armed forces and other capabilities in accordance with strategic plans. Meanwhile, the *operational level* is designated for campaigns and major operations in order to accomplish strategic objectives within theaters or areas of operations. Linking between tactics and strategies is done by establishing operational objectives needed to accomplish the strategic objectives, sequencing events to achieve the operational objectives, initiating actions and applying resources to bring about and sustain those events. Lastly, the *tactical level* involves battles and engagements, they are planned and executed to accomplish military objectives assigned to tactical units. The focus of this level is on the ordered arrangement and manoeuvre of combat elements in relation to each other and to the enemy in order to achieve combat objectives established by the operational level commander. In other words, the context of tactical operations is defined at the strategic and operational levels [9], [44].

Here, we focus on the planning process at the operational level. Planners at this level need to follow the Operational Art (OA) of using military forces. According to OA, the issues at this level include (1) identifying the military conditions or end-state that constitute the strategic objectives, (2) deciding the operational objectives that must be achieved to reach the desired end-state, (3) ordering a sequence of actions that lead to fulfilment of the operational objectives, and (4) applying the military resources (capabilities) allocated to sustain the desired sequence of actions. From this point onwards, we use the term mission planning to indicate planning at the operational level, unless otherwise stated.

There is no doubt that the planning process is based on a particular military doctrine. However, the main steps are similar among militarised forces. We will take the JMAP framework from Australian Defence Forces (ADF) [51] as an example

- Step 0 Initialization: including obtaining mission information (basically called Intelligence Preparation of the Battle-space - IPB)
- Step 1 Mission Analysis: Determining the objectives, available capabilities and other constraints for the mission. Also forming the commanders planning guidance for the next step.
- Step 2 Course Of Action (COA) Development: Developing the course of actions (ways to achieve the objectives with regards to the constraints)
- Step 3 COA Analysis: Comparing and analyzing COAs to obtain an optimal plan.
- Step 4 Decision & Execution: Deciding on the plan and executing it

Note that this is a repetitive process. Step 0 will be used to update information for all other steps. Once updated, the steps will be restarted for further analysis. If the time is available and the urgency is low, we will have a DEliberate Planning

(DEP) process. It is used to produce plans for contingencies and for later execution. When we need a plan for immediate action or in a very short time with a high urgency, we will have a crisis action planning (or immediate planning) process (CAP). These two types of planning are highly interrelated with each other. DEP produces the plans, while CAP uses these plans and adapt them to the current situations. In other words, CAP provides situation awareness.

For OA, it is also essential to define several key concepts. The concept of an end-state can be considered the final behaviour of the system when the system stops operating. For JMAP, end-state is defined as the set of conditions which will achieve the strategic objective. The national end-state is the set of desired conditions, incorporating the elements of national power, that will achieve the national objectives. The military end-state is the set of desired conditions beyond which the use of military forces is no longer required to achieve the national objectives. The military end-state for a mission planned at the operational level is defined by the command at the strategic level. This needs to be done in Step 1.

The next one is the concept of *Centre Of Gravity* (COG). For JMAP, COG is considered as the key characteristic, capability, or locality from which a military force derives its freedom of action, strength or will to fight. Another concept is *critical vulnerability* (CV). That is a characteristic or key element of a force that if it is destroyed, captured, or neutralized will significantly reduce the fighting capability of the force and its COG. Also the concept of *decisive point* (DP) needs to be considered. JMAP defines a DP as a major event that is a precondition to the successful disruption or negation of COG. A DP can be defined either in a time or geographical space. A mission plan might have many DPs. The line of DPs forms a path of attack or defeat to achieve the end-state. We also call it the *line of operations* - (LOP). Determining LOP is the most important component of the operational level planning. The sequence of operations needs to be done in order to achieve the end-state. Each operation or (action or task) is defined to take care one DP.

2) Mission planning problem: Determining COG, CV, and especially DPs is a challenge. It relies very much on the experience and knowledge of the staff and commanders. Further, given that these concepts are defined, finding LOPs is also a big deal. The scope of the problem means a large number of possible LOPs exist. The limitation of capabilities, synchronization of operations (the precedence relationship between operations), and time, make it very difficult to arrange sequences of tasks to achieves all DPs. From a computational point of view, it is valuable to quantify these concepts. Given the limitation of capabilities, precedence, and time , there is a need to schedule the tasks to obtain the optimal sequence.

Note that a task is defined as a tactical operation (or action) that a military force must do for achieving a DP. From the previous analysis, we can see that modeling tasks is a very important component. Quantitatively, a task usually has

- *A set of pre-conditions*: Defining operational conditions for a task that need to be achieved before commencing it. It might be derived from the defined relationship between tasks.
- *A set of effects*: it is usually defined by the DP.
- *A duration for execution*: A task can not be executed without a time limit in order to synchronize with other tasks.
- *A set of required capabilities*: This might be equipment, weapon, vehicles or troops.

Based on this definition, the planning problem can be transformed as follows

Parameters

- A set of tasks (a decomposition of the mission).
- A set of synchronization constraints for tasks.
- A limit on the capabilities available for the mission.

Objective

- Military end-state (that can be expressed as a set of conditions).

Outcome

- Plans that offer different lines of operations.

3) Dynamics and uncertainties: Dynamics and uncertainties are unavoidable factors for military missions. This is the nature of wars where enemies as well as environmental aspects are highly unpredictable. That is the reason for introducing the concept of the crisis action planning. One important requirements from the US Army is that the planning process needs to be continuous and adaptive to any changes. The presence of these factors, such as delaying in mission execution, failure of capabilities, or uncertain intuition of commanders on the relationship between operations of the mission, makes the task of mission planning more complex [44], [51], [28] with a large number of what-if scenarios that usually goes beyond the handling ability of human planners. Hence, there is a need for finding a robust and responsive mechanism in support planning staff.

There are many aspects that might be changed during the mission. Here, we describe three typical changes to demonstrate the concept, *the execution time of tasks*, *the availability of capabilities*, and *the relative relationship of tasks*. During the mission, there is no guarantee that a task will be completed in time. That might be caused by fatigue of the troops, equipment, logistics, or reinforcement by the enemy. Because of the limitation on the capabilities, if a task is late, there will be no return of the capabilities to do other tasks that are scheduled at that time. The question is how to adapt the current plan to deal with this change? It should be aware that any changes of the plan can cause a huge cost in terms of logistics and safety. Further disruption of capabilities might happen. This might be because of wounded troops or equipment damage. There is a need to adjust the plan to deal with this disruption. Finally, during the mission, the importance of a task might change due to reinforcement or changes by enemy forces. This might affect the relationship between tasks. Some tasks might be better to be executed before others. The staff and commanders have to figure out how to deal with this without too much cost (bearing in mind that the plan is already in use).

The case study looks at Re-active adaptation. Pro-active adaptation, sometimes called robustness, is not covered by the study.

4) Computational approaches for the mission planning problem: To date, a large amount of attention has been paid to applying computational approaches to deal with mission planning. An excellent review of computational approaches can be found at [9]. The computational techniques range from a formal method of Petri Net [52], decision theoretic method using Markov decision processes [4] to heuristic techniques such as tabu search [7], and evolutionary algorithms [50]. Moreover, mission planning problems can be formulated as a resource constrained project scheduling problem, a popular class of NP-hard problems, the use of heuristics and evolutionary algorithms should be the favorite choice.

Further, multi-objectivity is also addressed when introducing computational approaches to mission planning. In [50], the authors proposed a hierarchical planning system where the objectives can be the mission execution time, the cost of assets or the accuracy of executing tasks. Meanwhile, in [7] reliability was taken into account as as an objective together with the execution time. An interesting overview has been given in [13].

As stated in the above section, mission planning is a continuous and adaptive process. It constantly revises the plan over time to deal with changes. However, the issue of adaptation has been neglected in the area of computational decision support for mission planning. Most of the works focuses on addressing the aspect of robustness under uncertainties. In other words, they concentrated on the methodology of pro-activeness only. For examples, in [7], the authors used reliability as an additional objective to obtain a set of trade-off solutions; depending on the awareness of the decision makers, a solution is selected with an acceptable reliability with a hope that this solution can cope well with uncertainties. With a different perspective, the authors of [50] proposed to obtain a set of trade-off solutions, whenever a change happens, this set will be reviewed in order to select the best suitable one. For these approaches, the changes usually assumed happening with some bounds or with some anticipation. However, these assumptions might not always be satisfied; and further the changes over time might not be incremental. It can be the case that, once a change happened, all existing alternatives become infeasible. Hence there is a need to have a re-active adaptation mechanism for this case.

B. Test scenarios

We designed a military mission to validate our proposal. Note that this mission is aimed at providing an educational test only; it does not imply any particular mission. For this mission, the military is to face a major peacekeeping operation of protecting a troubled island. The strategic objective for this mission is to protect the newly installed government. The end state for this mission is the defeat of the insurgents. There will be three scenarios that the forces might have to deal with. They are described as follows.

For the first scenario (and we will call the equivalent APP as P1), the main available capabilities for this mission are limited to four types only including (exclude the landing facilities that are already provided conveniently):

Task ID	Duration	C1	C2	C3	C4
1	18	4	0	0	0
2	14	10	0	0	0
3	16	0	0	0	3
4	23	3	0	0	0
5	18	0	0	0	8
6	15	4	0	0	0
7	19	0	1	0	0
8	12	6	0	0	0
9	17	0	0	0	1
10	19	0	5	0	0
11	22	0	7	0	0
12	16	4	0	0	0
13	23	0	8	0	0
14	19	3	0	0	0
15	10	0	0	0	5
16	16	0	0	0	8
17	15	0	0	0	7
18	23	0	1	0	0
19	17	0	10	0	0
20	22	0	0	0	6
21	27	2	0	0	0
22	22	3	0	0	0
23	13	0	9	0	0
24	13	4	0	0	0
25	17	0	0	4	0
26	18	0	0	0	7
27	23	0	8	0	0
28	17	0	7	0	0
29	20	0	7	0	0
30	20	0	0	2	0

TABLE I
PROPERTIES OF TASKS

- 12 Light Mortar Batteries (C1)
- 13 Infantry Companies (C2)
- 4 C130s (C3)
- 12 Apache helicopters (C4)

After analyzing the mission, the commanders and staff concluded that the mission will have 30 tactical tasks including setting up bases/checkpoints, conducting surveillance by some special troops taken from infantry companies and by helicopters, securing the government, diplomatic missions, and foreigners, protecting some key infrastructures, attacking insurgent sites, and regular patrolling either in the cities or countryside. The precedence relationship between tasks is given in Figure 3. The requirements for these tasks are listed in Table I.

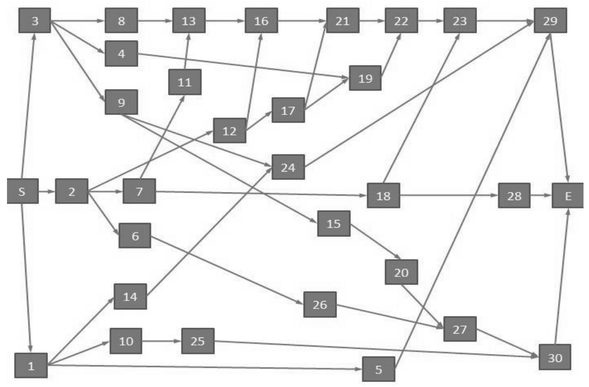


Fig. 3. Precedence network of tasks for P1.

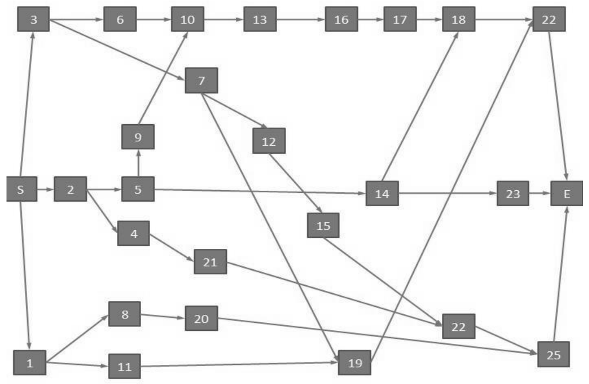


Fig. 4. Precedence network of tasks for P3.

For the second scenario (P2), there will be two more types of capabilities for the forces in dealing with the mission. That include 8 armor vehicles (C5) enforcing the troop mobilities and 4 companies (C6) of special forces tackling some dedicated tasks. The requests of these two types from the tasks are as follows [(0,0), (6,0), (3,2), (0,0), (0,0), (0,0), (2,0), (4,3), (0,0), (0,0), (0,0), (0,4), (0,0), (0,0), (0,0), (5,0),(0,0),(0,0), (3,0), (0,0), (0,0),(0,0),(0,0),(0,0),(0,0),(0,0), (3,0), (0,0),(0,0),(0,0),(0,0)]. These requests would constitute columns C5 and C6 of Table I for second scenario.

The last scenario (P3) includes only 25 main tasks. Five tasks (4, 5, 12 ,17, and 19) from the previous scenarios are no longer important, and hence they are removed from the list. The network of precedence is much simpler with 10 precedent of the arcs removed (see Fig. 4, noting the new indices of tasks). There are only four types of capabilities used in the first scenario.

Further, during the mission, the time delay in executing tasks is unavoidable. The intelligence source at that island is not highly reliable that cause the estimation of insurgents less accurate. This can be considered as a time-varying instance of the constraint in Eq. 3. Further, because of tropical weather, the failure of capabilities are highly expected. The dynamic version of the constraint in Eq. 4 is demonstrated via this. However, the current logistic supports at the island can help to quickly repair or reinforce the capabilities. Also, the precedence of the tasks is quite relative because of unreliable intelligence information. So changes of the precedence relationship is expected. Again it is used for a time-varying version of the third constraint in Eq. 5. So, we will have 9 problem instances being equivalent with three types of change (called Type 1, 2, and 3).

C. Parameter settings

From the problem description, we can see that the chromosome size is equal to the number of tasks. We used a population size of 40. The crossover and mutation rates were 0.9 and $1/n$ respectively. The size of the centroid set was 10 and the results were recorded after the tenth change. There is no particular reason for selecting these parameter's values, except they were set after a number of trials and they gave the most reliable performance. Each experiment was repeated for 30 times with different random seeds in the hope of eliminating the stochastic behavior caused by the random generator.

D. Validating methods

In the previous section, we described our evolutionary multi-objective approach using the CBA technique for starting a population after a change. In order to validate the proposed method, we also implemented it with three others adjusting from adaption for RCPS with a single objective:

- **The last population** - *LPOP*: For this approach, we use the last population obtained from the previous adaption period $P(t-1)$ (dealing with change at change $t-1$) as the initial population $P(t)$ to deal with the change t. Any solution that is infeasible with regard to the new conditions caused by the change will be randomly re-initialized. So, if $P(t-1)$ is the last population with size N, then $P(t)$ is defined as

$$P(t) = P1(t-1) + P'(t) \tag{13}$$

where $P1(t-1)$ is the set of $\overline{N1}$ solutions that are feasible under the new conditions caused by the change and $P'(t)$ is the set of $N - \overline{N1}$ newly randomly initialized.

- **The set of non-dominated solutions from the last population**- *NDLPOP*: Instead of using all individuals in the last population, we propose to use the non-dominated only. This will help to focus on the area of the best solutions only. The rest of the population will be randomly initialized to ensure diversity of the population at some degrees. $P(t)$ is defined as

$$P(t) = \overline{P(t-1)'} + P'(t) \tag{14}$$

where $\overline{P(t-1)}$ is the set of $\overline{N'}$ non-dominated solutions that are feasible under the new conditions caused by the change and $P'(t)$ is the set of $N - \overline{N'}$ newly randomly initialized.

- **Randomly initialized population** - *RI*: This method simply creates $P(t)$ by randomly initialization without caring any information in the past.

E. Results and discussion on behaviour of CBA

We use the first test scenario (or equivalently the dynamic instances of problem P1) to demonstrate the behaviour of the proposed approach. We look at the results after 10 changes from a run and take these results for our analysis. For the first instance (Type 1), the changes happened at time slots 1, 3, 7, 10, 13, 14, 17, 18, 19, and 22. A visualization of the plans can be found in Figures 5, 6, and 7 - the baseline schedule, the adapted schedule after the first change, and the adapted schedule after the tenth change respectively. At time zero, the baseline plan in Figure 5 indicated that task 2 was only one that needed to be executed first and it was followed by tasks 1, 3, and 7 at time 14. The objective values of the plan were (225 and 719.387). Note that tasks 1 and 2 can not be scheduled at the same time since both required 14 units of C1 while the maximum of C1 is 12. After a change at time 1 (note that task 2 was in-progress), it found 7 new adaptive plans. The plan with objective values of (194.517 and 721.451) was selected with an assumption that it is a suitable one from the commander's perspective. With this plan, task 3 was scheduled at time 1, as illustrated in Figure 6. The process continued until the change number 10. Again, we obtained 8 plans trading-off on time and cost. Note in Figure 7 that at this time tasks 2 and 3 were already completed and tasks 1, 4, 7, and 12 were in-progress. So their time was not affected by the adaptation process. The optimization process found a better solution after the change which prove the adaptability of the system. This can be seen by looking at Figures 5, 6, and 7. They show the new plan after every change is a non-dominated solution which implies that the new plans are better and so shows the system is adapting.

For the second instance (Type 2), the first change was at time 4 when only task 2 was in-progress (it started at time 0) and task 3 was about to start. After the change, we adapted the plan with the remaining tasks (including task 3). We obtained 7 new plans, assumed the plan with objective values of (194 and 1066.669) for execution. With this plan, tasks 1, 3, 4, 6, 7, 9, 10 , and 11 were kept for their original starting-times and the rest was adjusted with new starting-times. Finally, after the tenth change, we obtained new 4 plans for adaptation process and these plans can be submitted to the decision makers.

In the case of the third instance (Type 3), after the first change (at time 4), we obtained 10 trade-off solutions. The process continued in the manner described above. However, with the third instance the new adapted plan kept the same starting-times for only two tasks. This result shows the nature of the change caused by this type, it shows a large difference in terms of feasibility before and after a change.

Regarding the constraints of APP, we need to recall that the plans must be feasible against the precedence, time, and capability constraints. Also, from the obtained results, we observed that with Types 1 and 2, the solutions in the last population (before a change) were all infeasible to the time and capability constraint. The precedence is still preserved in all solutions. Meanwhile, this rate was (40, 40, 40, 0, 5, 40, 15, 31, 21, and 37) for Type 3 with regards to the precedence constraint (note that if the precedence is violated by a solution, this solution will not be considered further to the next constraints). However, this does not means that this type is easy, but in fact it is the hardest one since the sequence of indices (genotypes) requires a change while the first two do not.

It should be noted that the original solutions are still re-considered after every change. If they are still feasible, they will be included for consideration, and they might be excluded during the evolution process if they become dominated. The above results indicated that for our examples, the original solutions did not appear at the end in the non-dominated set as they were either infeasible or eliminated during evolution process.

Further, our approach relies on the use of centroids to boost the ability of the search. This depends a lot on the diversity of this set. It is worthwhile to look at the variation of centroids over the range of changes (see Figure 8). In this figure, we visualized the set of centroids for each type of changes (the first graph for Type 1, the second is for Type 2, and the last graph is for Type 3). The index axis has 30 values representing 30 values of a centroid, while the second axis is for the value of each element of the centroid. We can see from the figure that the set of centroids is quite diverse over all three types of dynamics. The location of the centroid moved. The nature of the change is also shown via this visualization where the changes for the first two types seemed to happen more with the lower index tasks than the higher-index ones. Meanwhile for Type 3, this seemed not the case, all tasks appeared quite similar. That is because of the effect of changes on Type 3 problems. A change of precedence between two tasks can cause the whole sequence to become infeasible and therefore the schedule. A change in Type 1 does not alter the skeleton of this sequence.

Diverse Pareto fronts can be seen in Figure 9. The figure shows a visualization of the non-dominated solutions obtained from all runs with different grey scale. The demonstrates the wide diversity of solutions obtained. It shows us quite diverse sets of non-dominated solutions spreading over two objectives: time and cost. It is important to have this diversity since we need to offer the decision makers alternatives for adapting against the change. Based on the capacity to afford the cost of the plan, the decision makers will select the final plan for the adaptation process. Further this indirectly affects the diversity of the centroid set that we use to guide the search. The above analysis has shown this matter.

The views of mission planning experts are naturally difficult to release. The views of mission planning experts are not covered because of confidentially issues.

F. Comparison of CBA to other methods

Understanding the behaviour of the method is important. However, it is essential to validate the performance of the obtained solutions. Here we discuss the relative performance of CBA in comparison to other methods (LPOP, NDLPOP, and RI) on all instances of all test problems. We compare them using the non-dominated plans obtained after the last change (after change No 10) for each of the 30 runs. Here we use the measure of the 'set coverage' - SC [18] to access the performance of these approaches. SC is determined between two sets A and B (SC(A,B)) by counting the number of solutions in B that are dominated by a solution in A:

$$SC(A,B) = \frac{|b \in B | \exists a \in A : a \preceq b|}{|B|} \tag{15}$$

where $a \preceq b$ indicates a dominates b. Obviously SC(A,B) is not necessarily equal to SC(B,A).

The mean values and standard errors from 30 runs are reported in Table II for all methods and all problems. It is clear that CBA obtained quite consistent results on all problems. It performed well in comparison to the others, for all nine problem-instances. We will look in more detailed at each problem in the following paragraphs.

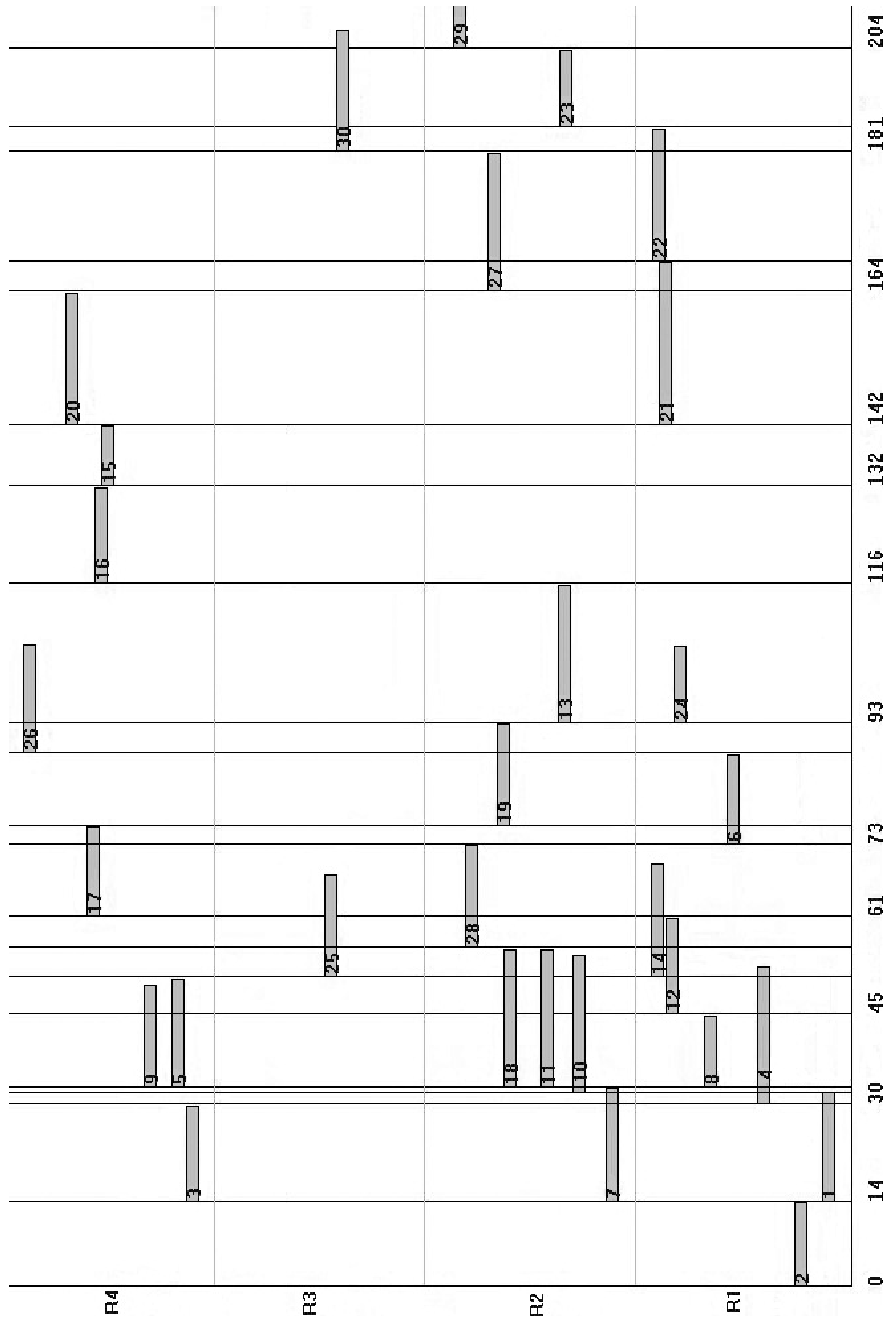


Fig. 5. The baseline schedule for P1. The horizontal axis is the time; the numbers in the rectangular strips are the task numbers; rectangular length is the task duration; and *R*s are the resources.

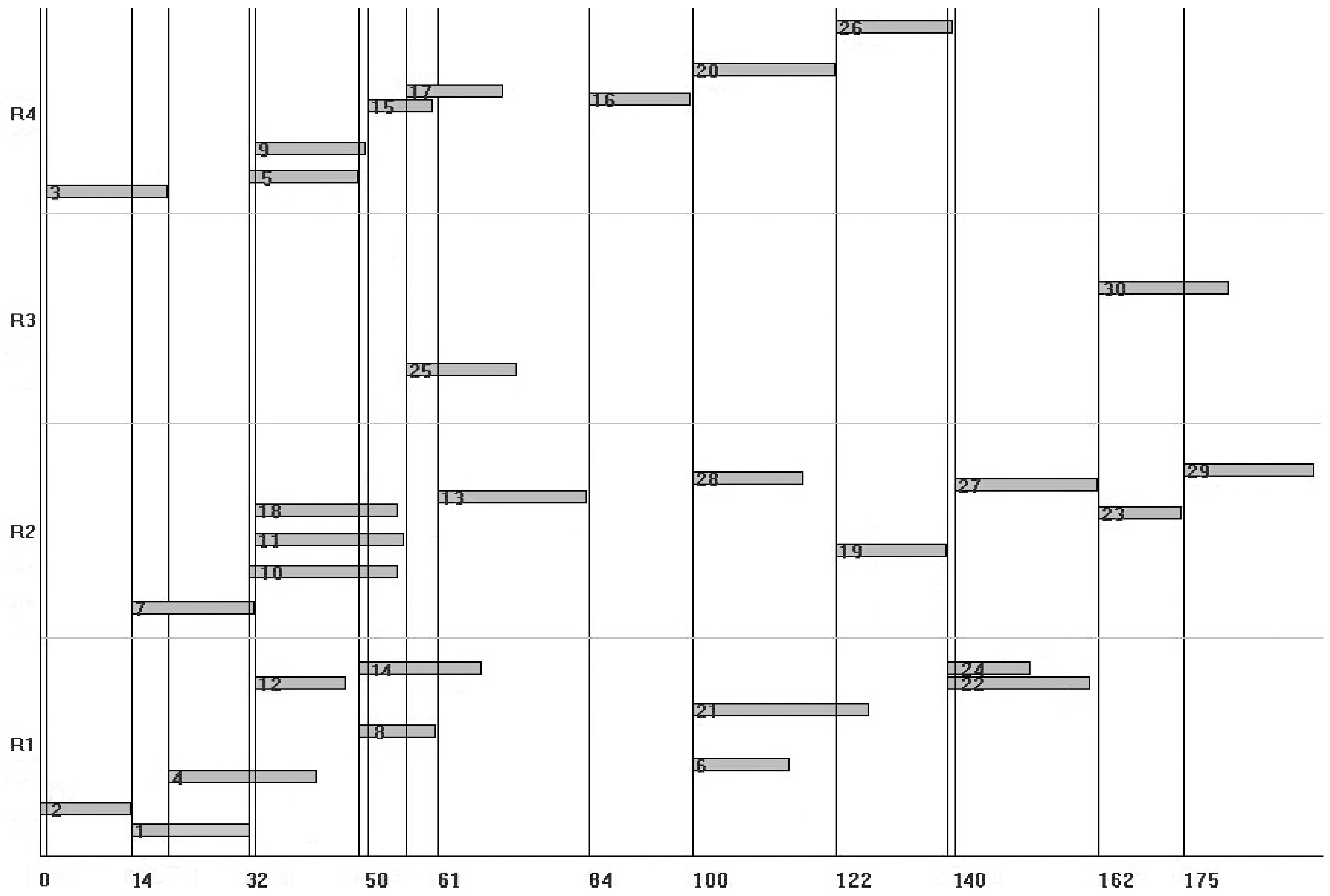


Fig. 6. In case of Type 1 and P1: The adapted schedule after the fist change obtained by CBA. The horizontal axis is the time; the numbers in the rectangular strips are the task numbers; rectangular length is the task duration; and Rs are the resources.

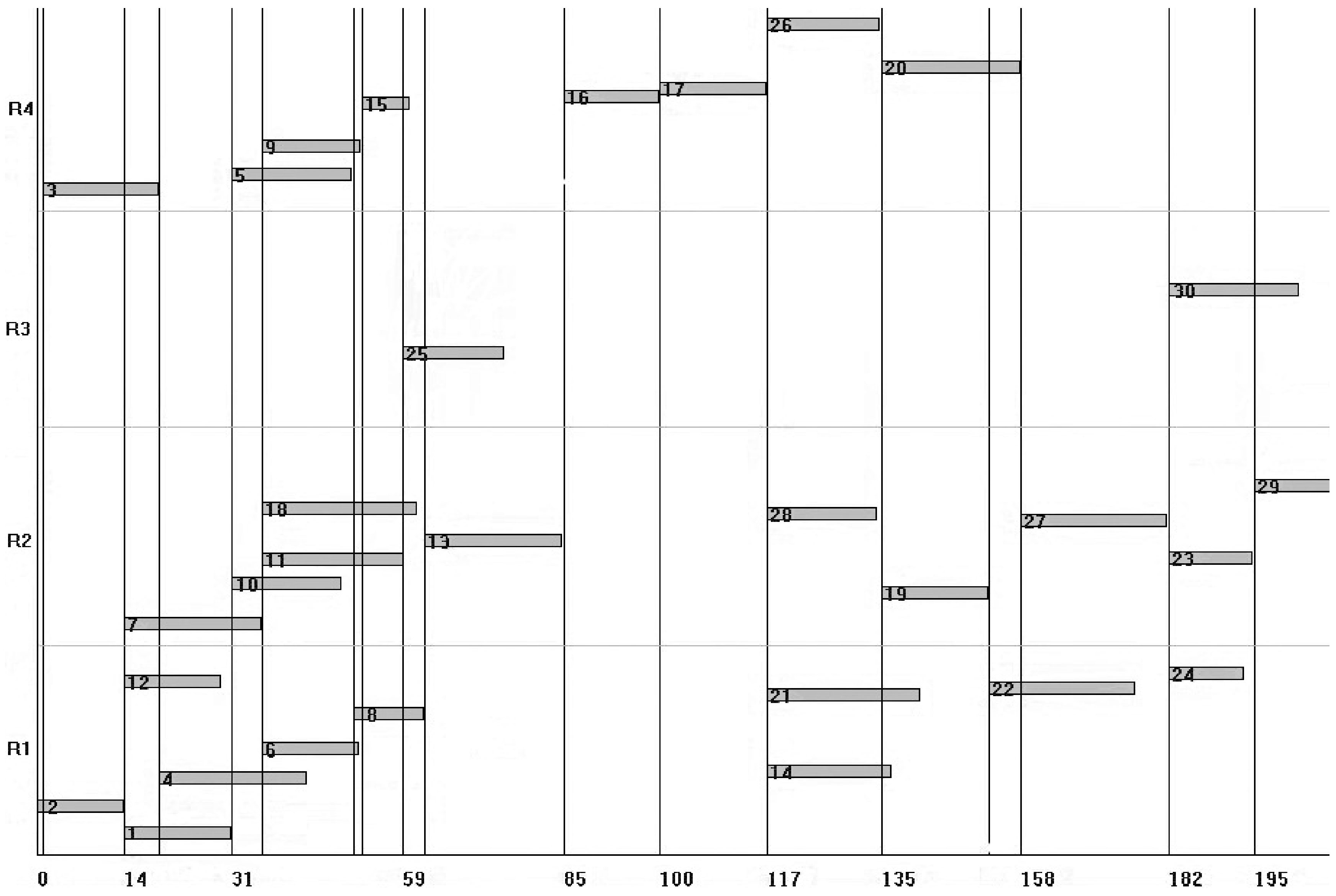


Fig. 7. In case of Type 1 and P1: The adapted schedule after the tenth change obtained by CBA. The horizontal axis is the time; the numbers in the rectangular strips are the task numbers; rectangular length is the task duration; and Rs are the resources.

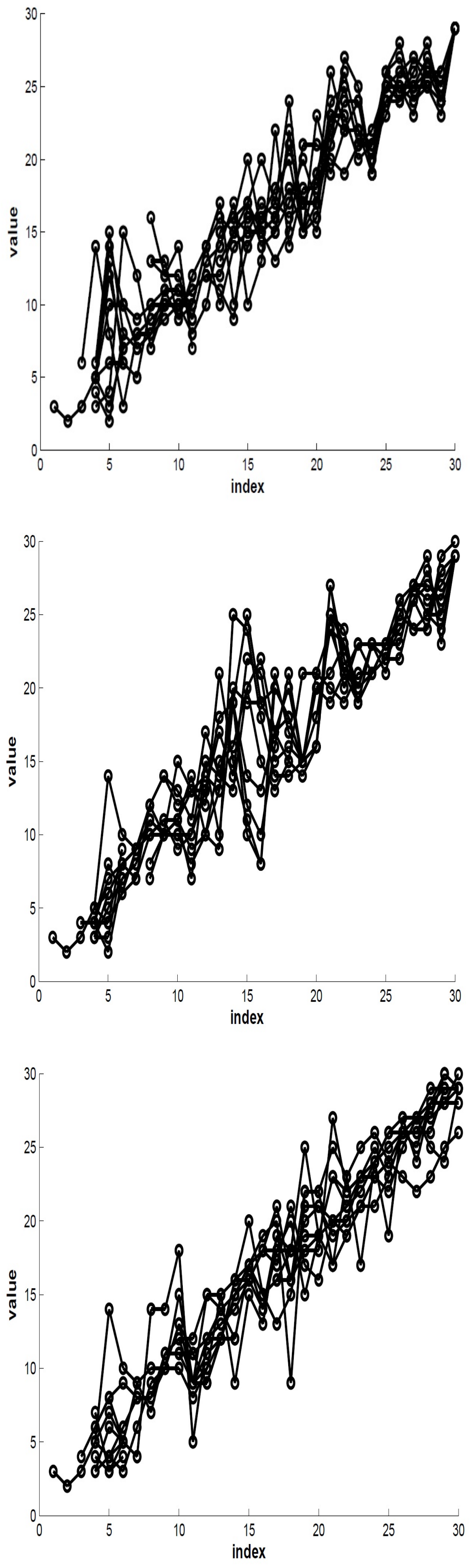


Fig. 8. Sets of 10 centroids (10 curves) obtained after ten changes and for each type of change : Each curve represents a centroid with 30 values (we have 30 tasks) - "index" means the vector of the centroid while the axis of "value" is for values of each index of the vector

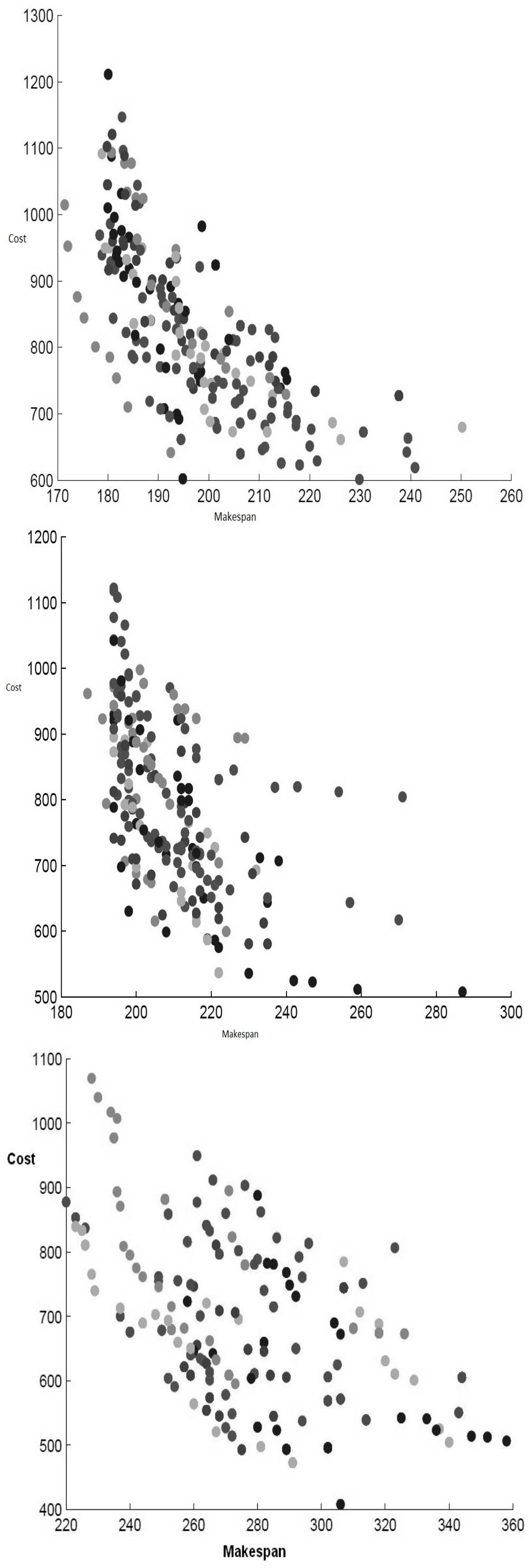


Fig. 9. Sets of non-dominated solutions over all 30 runs with two axes of time (horizontal) and cost (vertical): First graph is for Type 1, the second is for Type 2, and the last one is for Type 3

In the case of P1, for the first type, it obtained values of SC around 0.45 against all other methods (SC(CBA,X), where X is LPOP, NDLPOP or RI), while the reverse figures are just (0.325, 0.303, and 0.341) for LPOP, NDLPOP and RI respectively (SC(X,CBA), with X is either LPOP, NDLPOP or RI). Similarly, for P12 CBA obtained better SC values, that is (0.438, 0.437, and 0.556) in comparison to (0.345, 0345, and 0.195) respectively for LPOP, NDLPOP and RI. Meanwhile the third type showed an obvious difficulty for methods in finding non-dominated solutions. However, CBA still obtained much better SC values of (0.488, 0.488, and 0.458) in contrast to (0.214, 0.214, and 0.228) gained by others. The above finding provides strong support for our hypothesis of using centroids of non-dominated sets within the adaptation process. This information helped the search process quickly find a new area of optima.

With regards to the other methods, it seems that the use of the last population (LPOP) is a reasonable way to deal with change. However, it is not enough to effectively guide the search in general. With changes that have a significant impact (i.e. make the whole population severely infeasible), LPOP does not offer a good starting point for the adaptation process. That is why in comparison to RI, LPOP was better than RI for Types 1 and 2, but it was worse for the third type. As indicated above, for the third type, a single change might cause all of the current population to become infeasible, and if so the use of LPOP does not make much sense. RI was unstable, as it was inferior to LPOP in almost all cases. This is expected since the initial population is randomly re-initialized without any past information. Meanwhile, LPOP and NDLPOP had quite similar results. That is because for our test cases, the set of obtained non-dominated solutions occupied almost all the last population, especially in the case of Type 3, the last population was entirely filled by non-dominated solutions.

Further the results indicate the degree of difficulty of changing types. For Types 1 and 2, the changes did not make the sequence of tasks infeasible in terms of precedence constraint. The problem of adaptation is to find alternatives that give a suitable cost of adjusting capabilities. However, for Type 3, the sequence can become infeasible with even a single flip of the precedence network. This is reflected in the obtained results. For Type 3, the SC rates obtained against CBA were less than the other two (around 0.2 while 0.3 for the other types).

The findings are similar in the case of problem P2. Note that for the second scenario, the troops are supported with two additional resource types and P2 is formed with different requests for C5 and C6. Also, due to the new support, the cost values are re-adjusted. Again, RI still had the worst performance. It was better than LPOP and NDLPOP only in the case of Type 3 (0.394-0.273 for both LPOP and NDLPOP - in this case, LPOP and NDLPOP had the same performance). Again, this shows the difficulty from Type 3 for LPOP and NDLPOP as indicated in the case of problem P1. Meanwhile CBA obtained better performance overall with the exception of Type 2, where LPOP and NDLPOP were better.

However, for the the last problem (P3), the difficulty caused by changes was decreased since we dropped a significant number of tasks. Especially, for Type 3, 10 arcs were deleted from the precedence network. In other words, P3 is much simpler than P1 and P2. This might explain why RI gains in comparison to others. And the results have shown this. RI was better than the others for Types 1 and 2. However, for Type 3, RI was still unable to compete. The nature of Type 3 really made it hard for un-guided approaches (RI, LPOP and NDLPOP). Once more, CBA was much better for this type. For Types 1 and 2, CBA outperformed LPOP and NDLPOP.

To get more concrete support for CBA, we calculated the SC values of the non-dominated sets for several changes around the current being-considered change (change number 10). We started from the fifth change to allow the set of centroids had enough information until the change number 12. The average values of SC obtained after adapting to these changes were given in Table III. It turns out that the results firmly support the above finding. Again, CBA was consistently better than the others for almost all problem instances, especially for the most difficult one: Type 3. Meanwhile LPOP, NDLPOP and RI all had inconsistent behaviour. This clearly strengthens our preposition, that the use of centroid type past information certainly offers significant advantages for searching under new conditions caused by changes, without the need for any complex and costly mechanism.

G. *Effects of the last population on CBA*

A question that might be of interest with CBA is why we use the last population as a part of our new initial population to adapt against the change instead of random initialization? Recall that after using the set of centroids to derive some offspring, the remaining solutions will be adapted from the last population. To ease this concern, we will compare the performance of CBA with the original design and CBA using the randomly-initialized solutions (we call it CBAR). For CBAR, the remaining solutions will be re-initialized instead of adapting the the last population. Again the values of SC for both CBA and CBAR were reported in Table IV after the change No 10 and in Table V for the average from changes around change No. 10.

From the tables, we can see that CBA has slightly better performance than that of CBAR for either the 10th change or the average. For P1, CBA was better than CBAR for Types 2 and 3 while slightly worse for Type 1. It is similar for the case of P2 when comparing CBA and CBAR where CBA was better CBAR for all types in average (Table V). Meanwhile, for P3 CBAR was slightly better than CBA on Type 1 and worse on Types 2 and 3. This means that the use of the last population still makes more sense than a completely random re-initialization. For the difficult changes (like type 2 and 3), the random re-initialization does not help. This is also consistent with our above finding in the previous section between LPOP and RI.

VII. Conclusion and future work

In this paper, we provide an overview on the use of EAs for solving DOPs. Via this overview, we identify the essential need for an adaptation process in dealing with DOPs in real-world scenarios. We proposed a novel evolutionary multi-objective

Problems	Types		CBA	LPOP	NDLPOP	RI
P1	1	CBA	NA±NA	0.458±0.070	0.479±0.072	0.452±0.069
		LPOP	0.325±0.067	NA±NA	0.083±0.037	0.389±0.073
		NDLPOP	0.303±0.067	0.040±0.015	NA±NA	0.350±0.068
		RI	0.341±0.069	0.376±0.075	0.400±0.074	NA±NA
	2	CBA	NA±NA	0.438±0.063	0.437±0.063	0.556±0.058†
		LPOP	0.345±0.074	NA±NA	0.021±0.012	0.532±0.068†
		NDLPOP	0.345±0.061	0.010±0.006	NA±NA	0.532±0.068
		RI	0.195±0.053†	0.283±0.066†	0.286±0.066	NA±NA
	3	CBA	NA±NA	0.488±0.077†	0.488±0.077†	0.458±0.082†
		LPOP	0.214±0.068†	NA±NA	0.014±0.010	0.235±0.062
		NDLPOP	0.214±0.068†	0.014±0.010	NA±NA	0.235±0.062
		RI	0.228±0.066†	0.409±0.073	0.409±0.073	NA±NA
P2	1	CBA	NA ±NA	0.432±0.057	0.416±0.056	0.493±0.066
		LPOP	0.387±0.055	NA ±NA	0.023±0.013	0.493±0.063
		NDLPOP	0.405±0.056	0.038±0.026	NA ±NA	0.512±0.066
		RI	0.323±0.057	0.290±0.059	0.296±0.058	NA ±NA
	2	CBA	NA ±NA	0.288±0.055	0.301±0.056	0.441±0.066
		LPOP	0.481±0.075	NA ±NA	0.037±0.028	0.549±0.070†
		NDLPOP	0.469±0.074	0.035±0.027	NA ±NA	0.535±0.072
		RI	0.328±0.058	0.273±0.060†	0.288±0.063	NA ±NA
	3	CBA	NA ±NA	0.436±0.066	0.436±0.066	0.353±0.069
		LPOP	0.254±0.065	NA ±NA	0.000±0.000	0.273±0.069
		NDLPOP	0.254±0.065	0.000±0.000	NA ±NA	0.273±0.069
		RI	0.336±0.070	0.394±0.068	0.394±0.068	NA ±NA
P3	1	CBA	NA ±NA	0.299±0.070	0.299±0.070	0.289±0.075
		LPOP	0.271±0.068	NA ±NA	0.028±0.024	0.380±0.082
		NDLPOP	0.271±0.068	0.028±0.024	NA ±NA	0.380±0.082
		RI	0.483±0.078	0.488±0.084	0.488±0.084	NA ±NA
	2	CBA	NA ±NA	0.475±0.089	0.480±0.090	0.370±0.076
		LPOP	0.316±0.079	NA ±NA	0.008±0.033	0.333±0.067
		NDLPOP	0.313±0.079	0.004±0.004	NA ±NA	0.326±0.068
		RI	0.451±0.070	0.450±0.081	0.483±0.081	NA ±NA
	3	CBA	NA ±NA	0.288±0.076	0.321±0.079	0.399±0.074
		LPOP	0.258±0.068	NA ±NA	0.008±0.028	0.451±0.085
		NDLPOP	0.239±0.068	0.000±0.000	NA ±NA	0.423±0.083
		RI	0.228±0.069	0.249±0.074	0.249±0.074	NA ±NA

TABLE II
THE VALUES OF SC OBTAINED AFTER ADAPTING TO THE CHANGE NO. 10. SYMBOL † IS USED TO SHOW THAT THE DIFFERENCE BETWEEN TWO APPROACHES ARE STATISTICALLY DIFFERENT (USING T-TEST WITH 0.05 LEVEL OF SIGNIFICANCE)

approach for adaption in dynamic environments using scheduling and planning as test beds. We investigate a special class of planning problems called Adaptive Planning Problem (APP). For this problem, decision makers are expected to prepare adaptive plans to deal with any changes that might happen during execution of a selected plan. The question is that given the current being-used plan, how to generate the new adaptation policy that can satisfy both objectives: keeping the execution within its time-line while maintaining the less cost of adjusting?

We adapt the current plan in a reactive-style using an evolutionary algorithm. For any task, which is already executed or in progress, it will not be scheduled again. In this way, the rescheduling process will be smaller and simpler over time since the number of tasks to be scheduled decreases. To assist the decision making, we use the second objective as an additional indication to select a new adapted plan. A set of plans are obtained trading-off between time and cost of re-allocating the capabilities. Further, since the change might happen frequently during the mission, it should be worthwhile to use the the results obtained in the past to initialization of the first population in order to speed up the adaptation process. We introduced the concept of centroid of the non-dominated set to this context. Each centroid presents the area of optimal obtained by our approach and a set of centroids will keep track all areas of optima in the past. The initialization of the population dealing with the effects of a change will be guided by a a set of centroids. This helps to speed up the search process toward the new area of optima setup by the change.

A case study based on a military mission was used to validate our approach. Three different scenarios were used to analyze the performance of the proposed approach. Also, three types of changes are proposed to the mission planning problem including the execution time variation, the failure of

Problems	Types		CBA	LPOP	NDLPOP	RI
P1	1	CBA	NA±NA	0.404±0.009	0.409±0.009	0.407±0.007
		LPOP	0.355±0.012	NA±NA	0.056±0.005	0.391±0.008
		NDLPOP	0.344±0.010	0.036±0.002	NA±NA	0.377±0.008
		RI	0.376±0.007	0.379±0.006	0.383±0.006	NA±NA
	2	CBA	NA±NA	0.426±0.008†	0.426±0.008†	0.470±0.009†
		LPOP	0.357±0.008†	NA±NA	0.035±0.002	0.470±0.008
		NDLPOP	0.355±0.008†	0.028±0.002	NA±NA	0.469±0.008
		RI	0.292±0.009†	0.292±0.006	0.294±0.007	NA±NA
	3	CBA	NA±NA	0.438±0.013†	0.438±0.013†	0.386±0.015
		LPOP	0.310±0.012†	NA±NA	0.007±0.001	0.246±0.006
		NDLPOP	0.289±0.010†	0.007±0.001	NA±NA	0.246±0.006
		RI	0.310±0.012	0.329±0.015	0.329±0.015	NA±NA
P2	1	CBA	NA ±NA	0.410±0.010	0.399±0.009	0.490±0.010†
		LPOP	0.366±0.003	NA ±NA	0.020±0.002†	0.484±0.007
		NDLPOP	0.381±0.003	0.035±0.001†	NA ±NA	0.502±0.007
		RI	0.339±0.010†	0.326±0.007	0.322±0.007	NA ±NA
	2	CBA	NA ±NA	0.320±0.009†	0.319±0.011	0.437±0.010†
		LPOP	0.470±0.013†	NA ±NA	0.030±0.004	0.501±0.013
		NDLPOP	0.471±0.015	0.038±0.004	NA ±NA	0.503±0.014
		RI	0.368±0.006†	0.319±0.010	0.316±0.011	NA ±NA
	3	CBA	NA ±NA	0.373±0.016†	0.373±0.016	0.343±0.012
		LPOP	0.301±0.010†	NA ±NA	0.006±0.001	0.269±0.012
		NDLPOP	0.301±0.010	0.006±0.001	NA ±NA	0.269±0.012
		RI	0.352±0.012	0.342±0.011	0.342±0.011	NA ±NA
P3	1	CBA	NA ±NA	0.383±0.011†	0.383±0.011†	0.249±0.013†
		LPOP	0.228±0.005†	NA ±NA	0.023±0.003	0.284±0.016†
		NDLPOP	0.228±0.005†	0.008±0.002	NA ±NA	0.284±0.016
		RI	0.560±0.012†	0.532±0.011†	0.532±0.011	NA ±NA
	2	CBA	NA ±NA	0.419±0.011†	0.421±0.010†	0.307±0.020†
		LPOP	0.253±0.010†	NA ±NA	0.023±0.003†	0.225±0.020†
		NDLPOP	0.253±0.010†	0.008±0.002†	NA ±NA	0.219±0.019†
		RI	0.509±0.015†	0.515±0.030†	0.525±0.029†	NA ±NA
	3	CBA	NA ±NA	0.290±0.004	0.320±0.004	0.366±0.013†
		LPOP	0.313±0.013	NA ±NA	0.039±0.001†	0.334±0.013
		NDLPOP	0.286±0.014	0.009±0.001†	NA ±NA	0.311±0.012
		RI	0.263±0.009†	0.293±0.009	0.293±0.009	NA ±NA

TABLE III

AVERAGE VALUES OF SET-COVERAGE OBTAINED FROM ADAPTATION OVER A RANGE OF CHANGES. SYMBOL † IS USED TO SHOW THAT THE DIFFERENCE BETWEEN TWO APPROACHES ARE STATISTICALLY DIFFERENT (USING T-TEST WITH 0.05 LEVEL OF SIGNIFICANCE)

capabilities, and the change of the precedence network. We also implement three other initialization techniques within the proposed evolutionary approach. The obtained results clearly showed the good performance of centroid-based technique on dealing with the effects of changes.

Contributions are as follows:

- A survey and classification on dealing with dynamic environments. We first pinpoint three types of change, and then classify two categories of dealing with dynamic factors: tracking optima and adapting the current solutions to the change.
- A new method CBA that exploits multi-objectivity to facilitate the adaptation process.

For future work, we plan to investigate more complex instances of APP and with the use of a more complex simulation that allows us to present the factor of human in the loop. This human factor could be simulated by holons; defined as autonomous and self-reliant agents in [50] which models battlefield scenario. Application of CBA to the evolutionary process involved in the task and resource assignment of these agents could enhance computational speed. A comparison with non-EA type methods would improving our understanding of Adaptation in Dynamic Environments

ACKNOWLEDGEMENTS

This work was partially funded by the ARC Discovery Grant DP0985723, by grant N516 384734 from the Polish Ministry of Science and Higher Education (MNiSW) and by Le Quy Don University - Vietnam. The authors thank anonymous reviewers who provided us with excellent comments, which were incorporated in the final version of the paper.

Problems	Types		CBA	CBAR
P1	1	CBA	NA±NA	0.399±0.067
		CBAR	0.403±0.068	NA±NA
	2	CBA	NA±NA	0.435±0.061
		CBAR	0.313±0.062	NA±NA
	3	CBA	NA±NA	0.328±0.068
		CBAR	0.156±0.053	NA±NA
P2	1	CBA	NA±NA	0.443 0.067
		CBAR	0.360 0.058	NA±NA
	2	CBA	NA±NA	0.452 0.071
		CBAR	0.385 0.068	NA±NA
	3	CBA	NA±NA	0.229 0.062
		CBAR	0.318 0.067	NA±NA
P3	1	CBA	NA±NA	0.569 0.237
		CBAR	0.584 0.087	NA±NA
	2	CBA	NA±NA	0.498 0.076
		CBAR	0.413 0.075	NA±NA
	3	CBA	NA±NA	0.316 0.073
		CBAR	0.213 0.065	NA±NA

TABLE IV

THE VALUES OF SET COVERAGE OBTAINED BY CBA AND CBAR AFTER ADAPTING TO THE CHANGE NO 10.

Problems	Types		CBA	CBAR
P1	1	CBA	NA±NA	0.368±0.010
		CAR	0.396±0.014	NA±NA
	2	CBA	NA±NA	0.450±0.009
		CBAR	0.343±0.011	NA±NA
	3	CBA	NA±NA	0.323±0.006
		CBAR	0.261±0.009	NA±NA
P2	1	CBA	NA±NA	0.453 0.004
		CBAR	0.351 0.006	NA±NA
	2	CBA	NA±NA	0.478 0.009
		CBAR	0.348 0.006	NA±NA
	3	CBA	NA±NA	0.273 0.012
		CBAR	0.248 0.002	NA±NA
P3	1	CBA	NA±NA	0.282 0.023
		CBAR	0.596 0.013	NA±NA
	2	CBA	NA±NA	0.387 0.019
		CBAR	0.467 0.016	NA±NA
	3	CBA	NA±NA	0.286 0.011
		CBAR	0.264 0.017	NA±NA

TABLE V

AVERAGE SET-COVERAGE VALUES GAINED FROM ADAPTATION OVER A RANGE OF CHANGES

REFERENCES

[1] B. Abbasi, S. Shadrokh, and J. Arkat. Bi-objective resource-constrained project scheduling with robustness and makespan criteria. *Applied Mathematics and Computation*, 180:146–152, 2006.

[2] H. A. Abbass, R. Sarker, and C. Newton. PDE: A Pareto frontier differential evolution approach for multiobjective optimization problems. In *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 971–978, Seoul Korea, 2001. IEEE Service Center.

[3] D. Aberdeen, S. Thibaux, and L. Zhang. Decision-theoretic military operations planning. In *Proc. ICAPS*, pages 402–411. AAAI, 2004.

[4] D. Aberdeen, S. Thiébaux, and L. Zhang. Decision-theoretic military operations planning. In *Proc. ICAPS*, 2004.

[5] J. Alcaraz and C. Maroto. A robust genetic algorithm for resource allocation in project scheduling. *Annals of Operations Research*, 102(4):83–109, 2001.

[6] A. Azaron and R. Tavakkoli-Moghaddam. Multi-objective timecost trade-off in dynamic PERT networks using an interactive approach. *European Journal of Operational Research*, 180:11861200, 2007.

[7] L. Belfares, W. Klibi, N. Lo, and A. Guitouni. Multi-objective tabu search based algorithm for progressive resource allocation. *European Journal of Operational Research*, 177:1779–1799, 2007.

[8] J. Blazewicz, J.K. Lenstra, and A.H.G. Rinnooy Kan. Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Math*, 5:11–24, 1983.

[9] A. Boukhtouta, A. Bedrouni, J. Berger, F. Bouak, and A.Guitouni. A survey of military planning systems. In *The 9th ICCRTS Int. Command and Control Research and Technology Symposium*, Copenhagen, Denmark, 2004.

[10] J. Branke. *Evolutionary optimization in dynamic environments*. Kluwer Academic Publishers, Massachusetts USA, 2002.

[11] J. Branke, T. Kaußler, C. Schmidt, and H. Schmeck. A multi-population approach to dynamic optimization problems. In *Adaptive computing in design and manufacturing*. Springer, 2000.

[12] Lam T. Bui, Michael Barlow, and Hussein A. Abbass. A Multiobjective Risk-based Framework for Mission Capability Planning. *New Mathematics and Natural Computation, World Scientific*, 5(2):459–485, 2009.

[13] L.T. Bui and S. Alam. *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*. Information Science Publishing, 2008.

[14] L.T. Bui, J. Branke, and H.A. Abbass. Multiobjective optimization for dynamic environments. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 3, pages 2349–2356.

[15] J. A. Carruthers and Albert Battersby. Advances in critical path methods. *Operations research quarterly*, 17(4):359–380, 1966.

[16] H.G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Technical Report AIC-90-001, Naval Research Laboratory, 1990.

[17] C. A. C. Coello. Evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, 1(1):28–36, 2006.

[18] K. Deb. *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, New York, 2001.

[19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(2):182–197, 2002.

[20] E. Demeulemeester and W. Herroelen. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. *Management Science*, 38(12):1803 – 1818, 1992.

[21] A Drexl and A Kimms. Optimization guided lower and upper bounds for the resource investment problem. *Journal of Operations Reseach Society*, 52:340–351, 2001.

[22] D.E. Goldberg and R.E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 59–68, 1987.

[23] JJ Grefenstette. Genetic algorithms for dynamic environments. *Parallel problem solving from nature*, 2:137–144, 1992.

[24] S. Hartmann and R. Kolisch. Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 127(2):394–407, 2000.

[25] W. Herroelen and R. Leus. Project scheduling under uncertainty: Survey and research potentials. *European journal of operational research*, 165(2):289–306, 2005.

[26] K. S. Hindi, H. Yang, and K. Fleszar. An evolutionary algorithm for resource-constrained project scheduling. *IEEE Trans. on Evolutionary Computation*, 6(5):512–518, 2002.

[27] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments - a survey. *IEEE Trans. on Evolutionary Computation*, 9(3):303–317, 2005.

[28] R.H. Kewley and M.J. Embrechts. Computational military tactical planning system. *IEEE Trans. on Systems, Man, and Cybernetics, Part C*, 32(2):161–171, 2002.

[29] R. Kolishch and S. Hartmann. Experimental investigation of heuristic for resource constrained project scheduling: An update. *European Journal of Operational Research*, 174:23–37, 2006.

[30] J. Leung. *Handbook of scheduling: algorithms, models, and performance analysis*. Chapman & Hall/CRC, 2004.

[31] S.C. Lin, E.D. Goodman, and W.F. Punch. A genetic algorithm approach to dynamic job shop scheduling problems. In *Proceedings of the seventh International Conference on Genetic Algorithms*, pages 481–488, 1997.

[32] D. Merkle, M. Middendorf, and H. Schmeck. Ant colony optimization for resource-constrained project scheduling. *IEEE Trans. on Evolutionary Computation*, 6(4):333–346, 2002.

[33] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, London, UK, 3rd edition, 1996.

[34] Z. Michalewicz, M. Schmidt, M. Michalewicz, and C. Chiriac. *Adaptive business intelligence*. Springer-Verlag New York Inc, 2006.

[35] N. Mori, S. Imanishia, H. Kita, and Y. Nishikawa. Adaptation to changing environments by means of the memory based thermodynamical

genetic algorithms. In T.Bäck, editor, *Seventh Int. Conf. on Genetic Algorithms*, pages 299–306. Morgan Kaufmann, 1997.

[36] A. Nagar, J. Haddock, and S. Heragu. Multiple and bi-criteria scheduling: a literature survey. *European Journal of Operational Research*, 81:88104, 1995.

[37] D. Ouelhadj and S. Petrovic. A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, 12(4):417–431, 2009.

[38] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems (2nd Eds)*. Prentice Hall, NJ, 2001.

[39] C.L. Ramsey and J.J. Grefenstette. Case-based initialization of genetic algorithms. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 84–91, 1993.

[40] Michael R. Rose and George V. Lauder. *Adaptation*. Academic Press, 1996.

[41] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications:Proceedings of the First Int. Conf. on Genettic Algorithms*, pages 93–100, Hillsdale, New Jersey, 1985.

[42] R. Slowinski. *Advances in project Scheduling*, chapter Multiobjective project scheduling under multiple-category resource constraints. Elsevier, 1989.

[43] R.K. Ursem. Multinational GAs: Multimodal Optimization Techniques in Dynamic Environments. In *Proceedings of the Genetic and Evolutionary Computation Conf. (GECCO-2000)*, pages 19–26. San Francisco, CA: Morgan Kaufmann, 2000.

[44] US Department of the Army. FM 5-0: Army Planning and Orders Production, 2005.

[45] S. Van de Vonder, E. Demeulemeester, and W. Herroelen. A classification of predictive-reactive project scheduling procedures. *Journal of Scheduling*, 10(3):195–207, 2007.

[46] S. Van de Vonder, E. Demeulemeester, and W. Herroelen. Proactive heuristic procedures for robust project scheduling: an experimental analysis. *European Journal of Operational Research*, 189(3):723–733, 2008.

[47] Ana Viana and Jorge Pinho de Sousa. Using metaheuristics in multiobjective resource constrained project scheduling. *European Journal of Operational Research*, 120:359–374, 2000.

[48] M. Wineberg and F. Oppacher. Enhancing the GAs ability to cope with dynamic environments. In *Proc. Genetic Evol. Comput. Conf*, pages 3–10, 2000.

[49] S.D. Wu, R.H. Storer, and P.C. Chang. One-machine rescheduling heuristics with efficiency and stability as criteria. *Computers and Operations Research*, 20(1):1–14, 1993.

[50] F. Yu, F. Tu, and KR Pattipati. Integration of a Holonic Organizational Control Architecture and Multiobjective Evolutionary Algorithm for Flexible Distributed Scheduling. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 38(5):1001–1017, 2008.

[51] L. Zhang, L. Falzon, M. Davies, and I. Fuss. On relationships between key concepts of operational level planning. In *Proceedings of 5th International Command and Control Research and Technology Symposium*, 2000.

[52] Lin Zhang, Lars M. Kristensen, Chris Janczura, Guy Gallasch, and Jonathan Billington. A coloured petri net based tool for course of action development and analysis. In *CRPIT '02: Proceedings of the conference on Application and theory of petri nets*, pages 125–134. Australian Computer Society, Inc., 2002.

[53] G. Zhu, JF Bard, and G. Yu. Disruption management for resource-constrained project scheduling. *The Journal of the Operational Research Society*, 56(4):365–381, 2005.

[54] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou, D. T. Tsahalis, J. Periaux, K. D. Papailiou, and T. Fogarty, editors, *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100. Int. Center for Numerical Methods in Engineering (Cmine), 2001.

[55] E. Zitzler, L. Thiele, and K. Deb. Comparision of multiobjective evolutionary algorithms: Emprical results. *Evolutionary Computation*, 8(1):173–195, 2000.