Calculus of Variations in Discrete Space for Constrained Nonlinear Dynamic Optimization*

Yixin Chen and Benjamin W. Wah
Department of Electrical and Computer Engineering
and the Coordinated Science Laboratory
University of Illinois, Urbana-Champaign
Urbana, IL 61801, USA
E-mail: {chen, wah}@manip.crhc.uiuc.edu
URL: http://manip.crhc.uiuc.edu

Abstract

In this paper, we propose new dominance relations that can speed up significantly the solution process of nonlinear constrained dynamic optimization problems in discrete time and space. We first show that path dominance in dynamic programming cannot be applied when there are general constraints that span across multiple stages, and that node dominance, in the form of Euler-Lagrange conditions developed in optimal control theory in continuous space, cannot be extended to that in discrete space. This paper is the first to propose efficient dominance relations, in the form of local saddle-point conditions in each stage of a problem, for pruning states that will not lead to locally optimal paths. By utilizing these dominance relations, we develop efficient search algorithms whose complexity, despite exponential, has a much smaller base as compared to that without using the relations. Finally, we demonstrate the performance of our algorithms on some spacecraft planning and scheduling benchmarks and show significant improvements in CPU time and solution quality as compared to those obtained by the existing ASPEN planner.

1 Introduction

A large variety of engineering applications can be formulated as constrained *dynamic optimization problems* with dynamic variables evolving over time. These problems can generally be classified into four types: a) continuous-time continuous-state problems, b) discrete-time continuous-state problems, c) continuous-time discrete-state problems, and d) discrete-time discrete-state problems. The first two classes are also known as *functional optimization problems* in classical calculus of variations and as *optimal control problems* in control theory. In the first two classes, variational techniques and calculus of variations are the major tools for solving continuous-space functional optimization.

In this paper, we are interested in the last class defined above, namely, *nonlinear constrained dynamic optimization problems in discrete space and time*:

$$\begin{array}{lll} \min\limits_{y} & J[y] = \sum\limits_{t=0}^{N} F(t, y(t), y(t+1)) & (1) \\ \text{such that} & G(t, y(t), y(t+1)) = 0, & \\ \text{and} & I[y] = 0, & \end{array}$$

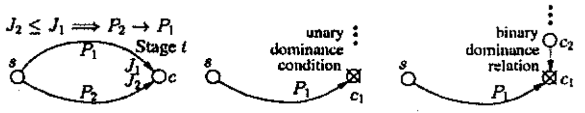
where $t = 0, 1, \cdots, N$. Here, $y_i(t)$ is the i^{th} discrete *dynamic state variable* in stage (time step) t; $y(t) = (y_1(t), \cdots, y_u(t))^T$ is a u-element *state vector* in discrete space $\mathcal{Y}$; $G = [G_1, \cdots, G_p]^T$ is a p-component vector of functions called the *Lagrange constraints* [2]; and $I = [I_1, \cdots, I_q]^T$ is a q-component vector of functions called the *general constraints*. Note that F, G and I are *not* necessarily continuous and differentiable in our definition. Further, although we have defined (1) with equality constraints, we show extensions later for handling inequality constraints.

A solution $y = (y(0), y(1), \cdots, y(N+1))$ to (1) consists of u discrete-time curves,[1] one for each dynamic state variable. Following conventional terminologies in continuous control theory, we call y a *bundle* (or a vector of curves) for both continuous- and discrete-time cases, and $J[y]$, a *functional* defined as a mapping from y to a numerical value.

*Research supported by National Aeronautics and Space Administration contract NCC 2-1230.

Proc. 14th IEEE International Conference on Tools with Artificial Intelligence, Nov. 2002.

[1]In this paper, we use the term "curve" uniformly for both continuous- and discrete-time cases. A discrete-time curve is typically named a "sequence."



$J_2 \leq J_1 \Longrightarrow P_2 \rightarrow P_1$
Stage t
s
P_1
P_2
J_1
J_2
c
unary dominance condition
P_1
c_1
binary dominance relation
c_2
c_1

a) Principle of optimality applied to feasible states
b) Termination of c_1 by unary dominance condition
c) Termination of c_1 by binary dominance relation

Figure 1. The application of path and node dominance in constrained minimization.

Many applications in discrete time and space can be formulated in (1). Ample examples exist in production planning and scheduling, chemical control processing, automated vehicle path planning, and action scheduling. These problems are characterized by multiple stages (time horizons) with a discrete search space in each stage, multiple Lagrange constraints across neighboring stages, and general constraints across more that two stages.

For example, planning problems in AI can be formulated in (1) using a discrete planning horizon, discrete state vectors representing positive and negative facts in each stage, and constraints representing preconditions and effects of actions. In that case, both space and time are discrete. Another example is in planning and scheduling spacecraft operations over a horizon [5]. The application involves finding a sequence of low-level commands from a set of high-level science and engineering goals, such as spacecraft operability constraints, flight rules, hardware models, science experiment goals, and operation procedures, subject to parameter dependencies and temporal and resource constraints. In that application, a Lagrange constraint may relate state variables in adjacent stages, whereas a general constraint may represent the total fuel in the system. The formulation has a discrete planning horizon, mixed-integer variables, and procedural objective and constraint functions.

Existing methods either solve (1) directly using heuristic guided-search methods that search in discrete path space under the guidance of heuristic functions, or transform (1) into a discrete constrained optimization problem before solving it by existing constrained programming techniques. The performance of these methods depends heavily on the search heuristics used and are not guaranteed to find feasible bundles. There is little research on techniques for reducing the worst-case complexity of (1).

The problem formulated in (1) is a multi-stage search problem, whose complexity can be reduced by dominance conditions and relations. Such dominance can be classified into path dominance studied in dynamic programming and node dominance.

Using the concept of bundles defined above and based on the *Principle of Optimality*, path dominance has been employed in dynamic programming and its variants for finding optimal bundles of a special case of (1) without general constraints. Figure 1a illustrates the application of the principle to an intermediate feasible state c in a constrained

Table 1. Worst-case complexities and the nature of the conditions in dominance. (Only node dominance can be used in solving (1) with general constraints.)

Constraint Type	Without General Constraints		With General Const. (W/O Path Dominance)	
Dominance Used	With Path Dominance	With Path & Node Dom.	W/O Node Dominance	With Node Dominance
Complexity	$O\left(N\|\mathcal{Y}\|^2\right)$	$O\left(N\|\mathcal{Y}\| + \sum_{i=1}^{N-1} \|\mathbf{s}(i)\| \times \|\mathbf{s}(i+1)\|\right)$	$O\left(\|\mathcal{Y}\|^N\right)$	$O\left(N\|\mathcal{Y}\| \prod_{i=1}^{N} \|\mathbf{s}(i)\|\right)$
Condition	Necessary and sufficient		–	Necessary

minimization problem without general constraints $I[y] = 0$. Starting from initial state s, assume that bundle P_1 (*resp.* P_2) leads to intermediate feasible state c with objective value J_1 (*resp.* J_2, where $J_2 \leq J_1$), based on variables assigned in P_1 (*resp.* P_2). The principle states that, if c lies on the optimal bundle from s to the final state, then P_1 will not be part of the optimal bundle because P_1 is *dominated* by P_2 (path dominance $P_2 \rightarrow P_1$).

In applying path dominance, it is necessary to enumerate all possible feasible states in each stage. As there is no efficient algorithm for identifying only feasible states when constraints are nonlinear, we need to enumerate all possible states in each stage. To estimate this complexity, consider an $(N+2)$-stage search problem, with an initial state in the first stage, a final state in the last stage, and $y(t) \in \mathcal{Y}$ states in stage $t = 1, \cdots, N$, where $\mathcal{Y}$ is the discrete search space in each stage. The complexity to find the optimal bundle is $O\left(N|\mathcal{Y}|^2\right)$, which is polynomial in $|\mathcal{Y}|$. Table 1 summarizes the complexities and the types of condition involved.

When general constraints $I[y] = 0$ are present in (1), the Principle of Optimality cannot be applied to partial bundles because a dominated partial bundle may satisfy a general constraint that spans beyond the current stage, whereas a dominating partial bundle may not. Hence, an algorithm for finding the optimal bundle will need to enumerate all possible bundles across all stages, leading to a worst-case complexity of $O\left(|\mathcal{Y}|^N\right)$, which is exponential[2] in $|\mathcal{Y}|$.

In contrast to path dominance, dominance conditions and relations can be developed to prune nodes in order to reduce $|\mathcal{Y}|$ in each stage of a search. Such pruning can be achieved by developing unary conditions (for pruning c_1 in Figure 1b) or binary relations (using $c_2 \rightarrow c_1$ to prune c_1 in Figure 1c).

Assume that $\mathbf{s}(t) \subseteq \mathcal{Y}$, $t = 1, \cdots, N$, is the set of nodes not pruned by node dominance in stage t in the multi-stage example above and that general constraints $I[y] = 0$ in (1) are absent. In this case, an exhaustive search for finding the optimal bundle will first enumerate all nodes in stage t, $t = 1, \cdots, N$, and apply node dominance in

[2]The implicit assumption is that (1) is NP-hard; hence, the worst-case complexity to find an optimal bundle is exponential.

order to identify $\mathbf{s(t)}$. It then applies path dominance on the $|\mathbf{s(t-1)}| \times |\mathbf{s(t)}|$ pairs across adjacent stages in order to identify the best bundle. This leads to a worst-case complexity of $O\left(N|\mathcal{Y}| + \sum_{i=1}^{N-1} |\mathbf{s(i)}| \times |\mathbf{s(i+1)}|\right)$ (third column in Table 1), which is better than the worst-case complexity when path dominance is applied alone. Note that node dominance, when applied in conjunction with path dominance, is necessary as well as sufficient because dominating bundles involving dominating nodes satisfy the Principle of Optimality.

On the other hand, when general constraints $I[y] = 0$ are present in (1) and nodes in a stage are pruned by node dominance, it is not possible to assert that a dominating node will satisfy a general constraint. Hence, node dominance is only necessary but no longer sufficient for feasibility. Further, to find the optimal bundle, one needs to enumerate all possible combinations of bundles of $|\mathbf{s(t)}|$ nodes in stage $t = 1, \cdots, N$, leading to a worst-case complexity of $O\left(N|\mathcal{Y}| \prod_{i=1}^{N} |\mathbf{s(i)}|\right)$ (last column in Table 1). Here, we assume a worst-case complexity of $|\mathcal{Y}|$ in stage t for finding one of the $\mathbf{s(t)}$ nodes in an enumerated bundle. Although this complexity is exponential with respect to $|\mathbf{s(t)}|$, it is substantially less than $O\left(|\mathcal{Y}|^N\right)$ (fourth column in Table 1) when $|\mathbf{s(t)}| \ll |\mathcal{Y}|$.

Node dominance in Figure 1b has been applied in the calculus of variations in continuous space. Since the conditions there have only been proved to be necessary, a dominating node may not lead to feasible bundles, even when general constraints are absent.

There are no known node dominance conditions and relations that are applicable for solving (1). We address this issue in this paper by developing a set of binary node dominance relations for each stage of (1). Our dominance relations are based on the theory of Lagrange multipliers in discrete space [8] and represent an extension of the calculus of variations in discrete space. They are necessary and sufficient in the absence of general constraints, and are only necessary when general constraints are added.

We first show that the problem of finding a bundle for (1) without general constraints amounts to finding a saddle point in a discrete neighborhood of the bundle, where a neighborhood is the union of discrete neighborhoods in each stage. Since the neighborhood of a bundle is very large when the number of stages is large, it is hard to find feasible bundles within such a neighborhood. To this end, we develop the calculus of variations in discrete space that allows the search for a saddle point in the discrete neighborhood of a bundle to be reduced to the search of multiple local saddle points, one in each stage of the problem. Moreover, we show that the collection of local saddle points are necessary and sufficient for local optimality in the same way a single saddle point is required for the local optimality of a bundle.

Based on the theory, we develop DCV-CSA, a general search algorithm that uses constrained simulated annealing (CSA) [7] to look for local saddle points in each stage. CSA performs descents in the original variable subspace in each stage and ascents in the Lagrange-multiplier subspace in order to locate saddle points. Finally, we apply DCV-CSA to solve some spacecraft planning and scheduling benchmarks and report much better results than those found by the existing ASPEN planner [4].

2 Previous Work

2.1 Calculus of variations in continuous space

In classical variational theory, a fundamental continuous-time continuous-state functional optimization problem in its simplest form is defined as:

$$\min_y J[y] = \int_{t_0}^{t_1} F(t, y(t), y'(t))dt, \; y(t) \in R^u. \qquad (2)$$

Since it is generally difficult to seek conditions for an absolute minimum curve in a search space, the theory in the continuous case defines the concepts of *strong* and *weak relative minimum* [3]. The study led to a number of necessary conditions for a curve to be a weak relative minimum, among which the most important is the *Euler-Lagrange* condition stated below in its differentiated form.

Theorem 1. *Continuous-time continuous-state Euler-Lagrange condition* [3]. If bundle y is a weak relative minimum to (2), and if t is any point in $[t_0, t_1]$ where derivative $y'(t)$ exists, then the following Euler-Lagrange equation holds:

$$F_{y_i}(t, y(t), y'(t)) - \frac{\partial}{\partial t} F_{y_i'}(t, y(t), y'(t)) = 0. \qquad (3)$$

Analogous to variational theory in continuous time and space, calculus of variations in discrete time and continuous space was studied since the 1960s. A *fundamental functional optimization problem in discrete time and continuous state* is defined as:

$$\min_y J[y] = \sum_{t=0}^{N} F(t, y(t), y(t+1)), \; y(t) \in R^u. \qquad (4)$$

Such studies led to discrete-time Euler-Lagrange equations, maximum principle, Noether Theorem, and extensions to integrable systems.

Although these results do not consider constraints, side constraints, including Lagrange constraints and isoperimetric constraints [2], can be included. Constraints are typically handled by using Lagrange multipliers to transform a constrained problem into an unconstrained one, and by applying the Euler-Lagrange condition on the Lagrangian function to get the Euler-Lagrange equations. The conditions obtained are only necessary but not sufficient, as the theory of Lagrange multipliers only yields necessary conditions for continuous optimization.

Unlike (2) and (4), constraints in general control problems [2] are introduced explicitly by a *control vector* and a *control function* governing the system. The major principle for such control problems is the Pontryagin minimum principle [3, 2] that can be derived by first treating the control function as a side constraint, transforming a general problem into a constrained fundamental problem, constructing a Lagrangian function using Lagrange multipliers, and finally solving the Euler-Lagrange equations.

The theory in continuous space cannot be used to solve (1) in discrete space when functions are not continuous and differentiable. To solve (1), we apply our theory of Lagrange multipliers for discrete constrained optimization [8] to (1) and show that a bundle satisfying the discrete-neighborhood saddle-point condition is necessary and sufficient for it to be a local-minimum bundle in its discrete neighborhood. We then establish the Euler-Lagrange equations for (1) and show that the saddle-point condition on a bundle can be partitioned in multiple saddle-point conditions, one for each stage of the problem.

2.2 Lagrangian theory on discrete constrained optimization

Consider a discrete equality-constrained nonlinear programming problem (NLP):

$$\begin{array}{lll} \min\limits_{x} & f(x) & \text{where } x \text{ is a vector of} \\ \text{subject to} & h(x) = 0, & \text{discrete variables.} \end{array} \qquad (5)$$

Here, $f(x)$ and $h(x) = [h_1(x), \cdots h_m(x)]^T$ are either analytic or procedural. We do not include inequality constraint $g(x) \leq 0$ in (5), as it can be handled easily by transforming it into an equivalent equality constraint $\max(g(x), 0) = 0$. Such conversions are not used in Lagrangian methods in continuous space because they are not differentiable at $g(x) = 0$. However, they do not pose any problem in our algorithms because we do not require their differentiability. Moreover, practical search algorithms employ greedy search and do not enumerate all possible neighborhood points.

To characterize solutions sought in discrete space, we define $\mathcal{N}(x)$, the *neighborhood* [1] of point x in discrete space $\mathcal{X}$, as a *finite* user-defined set of points $\{x' \in \mathcal{X}\}$ such that x' is reachable from x in one step, that $x' \in \mathcal{N}(x) \iff x \in \mathcal{N}(x)'$, and that it is possible to reach every other x'' starting from any x in one or more steps through neighboring points.

Point $x \in \mathcal{X}$ is a *discrete-neighborhood constrained local minimum* (CLM_{dn}) if it satisfies two conditions: a) x is a feasible point, implying that x satisfies $h(x) = 0$, and b) $f(x) \leq f(x')$ for all feasible $x' \in \mathcal{N}(x)$. Point $x \in \mathcal{X}$ is a *discrete-neighborhood constrained global minimum* (CGM_{dn}) if x is feasible and $f(x) \leq f(x')$ for all $x' \in \mathcal{X}$.

A *generalized Lagrangian function* of (5) is [8]:

$$L_d(x, \lambda) = f(x) + \lambda^T H(h(x)), \qquad (6)$$

where $\lambda = [\lambda_1, \cdots, \lambda_m]^T$ is a vector of continuous Lagrange multipliers, and H is a non-negative continuous transformation function satisfying:

$$H(y) \begin{cases} = 0 & \text{iff } y = 0, \\ \geq 0 & \text{otherwise.} \end{cases} \qquad (7)$$

Function H is easy to design; examples of which include $H(h(x)) = [|h_1(x)|, \cdots, |h_m(x)|]^T$ and $H(h(x)) = [\max(h_1(x), 0), \cdots, \max(h_m(x), 0)]^T$. Note that these transformations are not used in Lagrangian methods in continuous space because they are not differentiable at $h(x) = 0$. However, they do not pose problems here because we do not require their differentiability.

A *direction of maximum potential drop (DMPD)* defined in a discrete neighborhood of $L_d(x, \lambda)$ at point x for fixed λ is a vector that points from x to $x' \in \{x\} \cup \mathcal{N}(x)$ with the minimum L_d:

$$\begin{array}{rcl} \Delta_x L_d(x, \lambda) & = & x' - x = (x'_1 - x_1, \cdots, x'_n - x_n) \\ \text{where } x' & = & \operatorname*{argmin}\limits_{y \in \mathcal{N}(x) \cup \{x\}} L_d(y, \lambda). \end{array}$$

A *discrete-neighborhood saddle point* $SP_{dn}(x^*, \lambda^*)$ has the following property:

$$L_d(x^*, \lambda) \leq L_d(x^*, \lambda^*) \leq L_d(x, \lambda^*), \qquad (8)$$

for all $x \in \mathcal{N}(x^*)$ $\lambda \in R^m$. SP_{dn} is different from SP_{cn} (saddle point in continuous space) because it is defined using different neighborhoods.

Theorem 2. *First-order necessary and sufficient conditions for CLM_{dn}* [8]. A point in the discrete search space of (5) is a CLM_{dn} iff:

a) it satisfies the saddle-point condition (8); or
b) it satisfies the following discrete-space first-order conditions:

$$\Delta_x L_d(x, \lambda) = 0 \text{ and } h(x) = 0. \qquad (9)$$

Theorem 2 is stronger than its continuous counterpart that requires CLM_{cn} (CLM in continuous space) to be regular points and all functions to be differentiable in its first-order necessary conditions. In contrast, these are not

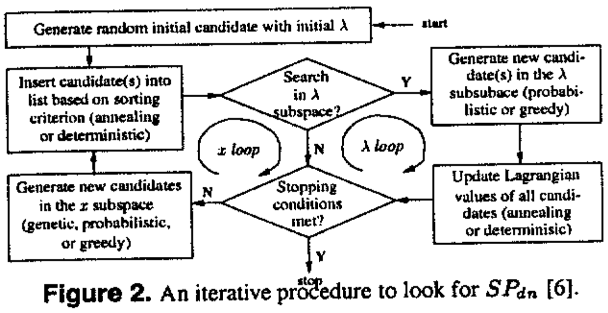

Generate random initial candidate with initial λ
start
Insert candidate(s) into list based on sorting criterion (annealing or deterministic)
Search in λ subspace?
Y
Generate new candidate(s) in the λ subsubace (probabilistic or greedy)
x loop
N
λ loop
Generate new candidates in the x subspace (genetic, probabilistic, or greedy)
N
Stopping conditions met?
Update Lagrangian values of all candidates (annealing or determinisic)
Y
stop

Figure 2. An iterative procedure to look for SP_{dn} [6].

required for CLM_{dn}. Further, the first-order conditions for continuous problems are only necessary and must be augmented by second-order sufficient condition, whereas the conditions in Theorem 2 are necessary as well as sufficient.

Based on the conditions in Theorem 2, Figure 2 depicts a general procedure to look for SP_{dn} [6]. The procedure consists of two loops: the x loop that updates the x variables in order to perform descents of L_d in the x subspace, and the λ loop that updates the λ variables of unsatisfied constraints in order to perform ascents in the λ subspace.

Various implementations have been developed based on the procedure, including the *discrete Lagrangian method* (DLM) [8], *constrained simulated annealing* (CSA) [7], and *anytime constrained simulated annealing* (ACSA) [6]. DLM entails greedy searches in the x and λ subspaces, deterministic insertions into the list of candidates, and deterministic acceptance of candidates until all constraints are satisfied. In contrast, CSA generates new probes randomly in one of the x or λ variables, accepts them based on the Metropolis probability if L_d increases along the x dimension and decreases along the λ dimension, and stops updating λ when all constraints are satisfied. In particular, using a logarithmic cooling schedule, CSA converges asymptotically to CGM_{dn} with probability one [7].

3 Calculus of Variations in Discrete Space

3.1 Basic definitions

Solutions to (1) cannot be characterized in ways similar to those of continuous-space problems with differentiable functions. In the latter class of problems, strong and weak relative minima are defined with respect to neighborhoods of open spheres with radius approaching zero asymptotically. Such a concept does not exist in problems with discrete variables.

To characterize solutions sought in discrete space, we define concepts on neighborhoods and constrained solutions similar to those in Section 2.2.

Definition 1. $\mathcal{N}_v(s)$, the *discrete neighborhood* [1] of state vector $s \in \mathcal{Y}$, is a finite user-defined set of states $\{s' \in \mathcal{Y}\}$ such that $s' \in \mathcal{N}_v(s) \Longleftrightarrow s \in \mathcal{N}_v(s')$. Further, for any $s^1, s^k \in \mathcal{Y}$, it is possible to find a finite sequence of state vectors $s^1, \cdots, s^k \in \mathcal{Y}$ such that $s^{i+1} \in \mathcal{N}_v(s^i)$, $i = 1, \cdots k-1$.

Definition 2. $\mathcal{N}_b^{(t)}(y)$, the *t^{th}-stage discrete neighborhood of bundle* y for given $\mathcal{N}_v(s)$ and all $t = 0, 1, \cdots, N+1$, is defined as follows:

$$\mathcal{N}_b^{(t)}(y) = \left\{ z \;\middle|\; z(t) \in \mathcal{N}_v(y(t)) \text{ and } z(i) = y(i), \right.$$
$$\left. i = 0, \cdots, N+1,\ i \neq t \right\}. \quad (10)$$

Intuitively, $\mathcal{N}_b^{(t)}(y)$ includes all bundles that are identical to y in all stages except t, where the state vector is perturbed to a neighboring state in $\mathcal{N}_v(y(t))$.

Definition 3. $\mathcal{N}_b(y)$, the *discrete neighborhood of bundle* y, is:

$$\mathcal{N}_b(y) = \bigcup_{t=0}^{N+1} \mathcal{N}_b^{(t)}(y) \quad (11)$$

Definition 4. Bundle y is a *discrete-neighborhood constrained local minimum* (CLM_{dn}) of (1) if a) y is feasible, implying that y satisfies all the constraints in (1), and b) $J[y] \leq J[z]$ for all feasible bundles $z \in \mathcal{N}_b(y)$.

Definition 5. A *generalized discrete Lagrangian function* of (1) is:

$$\begin{aligned} L_d(y, \lambda, \mu) \;\; &= \;\; J[y] + \sum_{t=0}^{N} \lambda^T(t) H\left(G(t, y(t), y(t+1))\right) \\ &\quad + \mu^T H\left(I[y]\right), \end{aligned} \quad (12)$$

where $\lambda(t) = [\lambda_1(t), \cdots, \lambda_p(t)]^T$, $t = 0, \cdots, N$, and $\mu = [\mu_1, \cdots, \mu_q]^T$ are vectors of Lagrange multipliers, and H is a non-negative continuous transformation function defined in (7).

Definition 6. A *discrete-neighborhood saddle point* $SP_{dn}(y^*, \lambda^*, \mu^*)$ of (1) is a point with the following property: For all $y \in \mathcal{N}_b(y^*)$ and $\lambda \in R^u, \mu \in R^q$,

$$L_d(y^*, \lambda, \mu^*) \leq L_d(y^*, \lambda^*, \mu^*) \leq L_d(y, \lambda^*, \mu^*) \quad (13)$$
$$\text{and } L_d(y^*, \lambda^*, \mu) \leq L_d(y^*, \lambda^*, \mu^*) \quad (14)$$

The following result follows from Definition 6 and Theorem 2.

Lemma 1. Bundle y in the discrete search space of (1) is a CLM_{dn} iff:

a) it satisfies the discrete-neighborhood saddle-point conditions (13) and (14); or

b) it satisfies the following discrete-space first-order conditions:

$$\Delta_y L_d(y, \lambda, \mu) = 0, \quad (15)$$
$$G(t, y(t), y(t+1)) = 0 \text{ and } I[y] = 0,$$

where $t = 0, 1, \cdots, N$.

Intuitively, Lemma 1 re-states the necessary and sufficient conditions in Theorem 2 when (1) is solved as a nonlinear equality-constrained NLP. These conditions, however, must be applied on a bundle as a whole and cannot help prune nodes in intermediate stages. In order to prune such nodes, we show in the next section the partitioning of these conditions into those that can be applied in each stage of a search.

3.2 Distributed necessary and sufficient conditions

Analogous to the Euler-Lagrange conditions in continuous space, we present in this section the partitioning of the discrete-neighborhood saddle-point condition in (13) into multiple saddle-point conditions, one for each stage. Before the main theorem is presented, we define the following to characterize the distributed properties of the solution space of (1).

Definition 7. The *t^{th}-stage distributed discrete Lagrangian function* of (1) for all $t = 0, \cdots, N+1$ is defined as:

$$\begin{aligned}
\Gamma_d(t, y, \lambda, \mu) &= F(t-1, y(t-1), y(t)) \\
&\quad + F(t, y(t), y(t+1)) \\
&\quad + \lambda(t-1) H(G(t-1, y(t-1), y(t))) \\
&\quad + \lambda(t) H(G(t, y(t), y(t+1))) + \mu H(I[y]) \\
\Gamma_d(0, y, \lambda, \mu) &= F(0, y(0), y(1)) \\
&\quad + \lambda(0) H(G(0, y(0), y(1))) + \mu H(I[y]) \\
\Gamma_d(N+1, y, \lambda, \mu) &= F(N, y(N), y(N+1)) \\
&\quad + \mu H(I[y]).
\end{aligned}$$

Definition 8. The *t^{th}-stage distributed direction of maximum potential drop* on a discrete neighborhood of y in stage t, $t = 0, \cdots, N+1$, is a vector that points from y to $y' \in \{y\} \cup \mathcal{N}_b^{(t)}(y)$ with the minimum $\Gamma_d(t, y, \lambda, \mu)$:

$$\Delta_{y(t)} \Gamma_d(t, y, \lambda, \mu) = y' - y \,,$$
$$\text{where } y' = \underset{z \in \{y\} \cup \mathcal{N}_b^{(t)}(y)}{\operatorname{argmin}} \Gamma_d(t, z, \lambda, \mu).$$

The main result is summarized as follows.

Theorem 3. *Distributed necessary and sufficient conditions for CLM_{dn}.* Bundle y in the discrete search space of (1) is a CLM_{dn} iff it satisfies either one of the following sets of conditions:

a) Discrete-neighborhood Euler-Lagrange equations (ELE_{dn}):

$$\Delta_{y(t)} \Gamma_d(t, y, \lambda, \mu) = 0, \quad t = 0, \cdots, N+1 \quad (16)$$
$$G(t, y(t), y(t+1)) = 0, \quad t = 0, \cdots, N \quad (17)$$
$$\text{and } I[y] = 0. \quad (18)$$

b) Distributed discrete-neighborhood saddle-point condition (DSP_{dn}):

$$\Gamma_d(t, y^*, \lambda', \mu^*) \le \Gamma_d(t, y^*, \lambda^*, \mu^*) \le \Gamma_d(t, y', \lambda^*, \mu^*), \quad (19)$$
$$\text{and } L_d(y^*, \lambda^*, \mu') \le L_d(y^*, \lambda^*, \mu^*), \quad (20)$$

for $t = 0, 1, \cdots, N+1$ and for all $y' \in \mathcal{N}_b^{(t)}(y^*)$, $\lambda' \in \mathcal{N}_b^{(t)}(\lambda^*)$, and $\mu' \in R^q$. Here,

$$\begin{aligned}
y' &= (y^*(0), \cdots, y^*(t-1), y'(t), y^*(t+1), \\
&\quad \cdots, y^*(N+1)) \\
\lambda' &= (\lambda^*(0), \cdots, \lambda^*(t-1), \lambda'(t), \lambda^*(t+1), \\
&\quad \cdots, \lambda^*(N)), \\
\mathcal{N}_b^{(t)}(\lambda^*) &= \{\lambda \mid \lambda(t) \in R^u \text{ and } \lambda(i \mid i \neq t) = \lambda^*(i)\},
\end{aligned}$$

where y' (*resp.* λ') represents y^* (*resp.* λ^*) perturbed in the t^{th} stage.

The theorem can be proved by examining the functional structure of the conditions in Lemma 1 and by decomposing the conditions in each stage. We do not show the proof in this paper.

There are several salient features of Theorem 3.

a) Unlike classical calculus of variations in continuous space, Theorem 3 does not require the differentiability or continuity of functions.

b) Conditions (16) and (17) in ELE_{dn} are the discrete-space counterpart of the continuous Euler-Lagrange equations, whereas (19) in DSP_{dn} indicates that finding a CLM_{dn} is equivalent to finding $N+2$ discrete-neighborhood saddle points, one in each stage. These conditions are the dominance relations for pruning nodes in a search. In particular, (19) is very easy to implement and is employed in the variational search algorithm presented next. Further, (18) and (20) refer to conditions on the general constraints that must be satisfied.

c) As is discussed in Section 1, the use of node dominance helps reduce the search complexity from $O\left(|\mathcal{Y}|^N\right)$ to $O\left(N|\mathcal{Y}| \prod_{i=1}^N |\mathrm{s(i)}|\right)$ (Table 1). As $|\mathrm{s(i)}|$, the number of points in $\mathcal{Y}$ of stage i that are feasible and are local minimum of Γ_d, is much smaller than $|\mathcal{Y}|$, node dominance helps reduce the base of the exponential complexity to a much smaller value.

```
1. procedure DCV-CSA
2.     Initialize starting point of $y$;
3.     Initialize $\lambda$, $\mu$ to 0;
3.     $\rho \leftarrow \rho_0$ ; /* vector $\rho$ controls the update of $\mu$*/
4.     repeat /* outer loop in iterative scheme */
5.         for $t = 0$ to $N + 1$ /* inner loop for $SP_{dn}$ */
6.             call CSA to find $SP_{dn}$ in stage $t$;
7.         end_for
8.         evaluate $I[y]$ and update $\rho$ if necessary;
9.         $\mu \leftarrow \mu + \rho^T H(I[y])$; /* ascents in $\mu$ subspace */
10.    until no change of $y, \lambda, \mu$ in an iteration;
11. end_procedure
```

Figure 3. DCV-CSA: an iterative procedure for finding points that satisfy DSP_{dn} in Theorem 3 using CSA [7] in Figure 2.

d) In a way similar to that described in Section 2.2, our theory handles inequality constraints by transforming $g(x) \leq 0$ into an equivalent equality constraint $\max(g(x), 0) = 0$.

3.3 Discrete-Space Variational Search

Figure 3 outlines DCV-CSA, a heuristic procedure for finding points that satisfy DSP_{dn} in Theorem 3. It consists of two parts, one looking for dominating nodes (local saddle points) in stage t that satisfy (19), and the other looking for local maxima in the μ subspace of $L_d(y, \lambda, \mu)$ that satisfy (20). It uses CSA in Figure 2 with default parameters to look for SP_{dn} in each stage, and, once all the stages have been examined, performs ascents of $L_d(y, \lambda, \mu)$ in the μ subspace if there were unsatisfied general constraints. It stops when no further improvements can be made in the y, λ and μ subspaces. In principle, the inner loop can use any algorithm that looks for SP_{dn}.

4 Experiments on ASPEN

We have tested DCV-CSA on several benchmarks on planning and scheduling of spacecraft operations, including CX1-PREF, OPTIMIZE, and PREF, that were designed originally for the ASPEN (Automated Scheduling and Planning Environment) planner [4] at the Jet Propulsion Laboratory. The CX1-PREF benchmark models the Citizen Explorer-1 satellite design and operation planning [9]. It has a problem generator that can generate problem instances of different number of satellite orbits.

Planning and scheduling of spacecraft operations involves generating a sequence of low-level spacecraft commands from a set of high-level science and engineering goals. Using discrete time horizons and a discrete state space, an ASPEN model encodes spacecraft operability constraints, flight rules, spacecraft hardware models, science experiment goals, and operations procedures in order to allow for the automated generation of low-level spacecraft sequences. It defines various types of constraints, including temporal constraints, decomposition constraints, resource constraints, state dependency constraints, and goal constraints, that may be procedural and not in closed form. In addition, it defines the quality of a schedule in a *preference score*, a weighted sum of multiple preferences (that may also be in procedure form) which the planner tries to optimize.

The ASPEN planner interleaves repair-based planning with preference-driven, incremental optimization. Using a rich collection of heuristic-repair and optimization actions, ASPEN generates promising search directions in the solution space of schedules. In our experiments, ASPEN alternates between a repair phase with unlimited iterations and an optimization phase with 200 iterations.

We have integrated DCV-CSA into ASPEN in order to test its performance. Since ASPEN has discrete time horizons, we can treat each time point as a discrete stage or collapse multiple adjacent time points into a single stage. In our implementation, we divide the time horizon evenly into 100 stages (although one may improve the performance further by adjusting the boundary of stages at run time in order to balance the activities across different stages more evenly). In performing descents of Γ_d in a stage, we choose probabilistically among ASPEN's repair and optimizations actions, select a random feasible action at each choice point, apply the selected action to the current schedule, and accept the new schedule based on the Metropolis probability in CSA.

Figure 4 compares the performance of our proposed approach against that of ASPEN on four problems: CX1-PREF with 8 orbits, CX1-PREF with 16 orbits, OPTIMIZE, and PREF. For each benchmark, we plot the the quality of the best feasible schedule found with respect to the number of search iterations. The results show that DCV-CSA is able to find bundles of the same quality one to two orders faster than ASPEN and much better bundles when it converges.

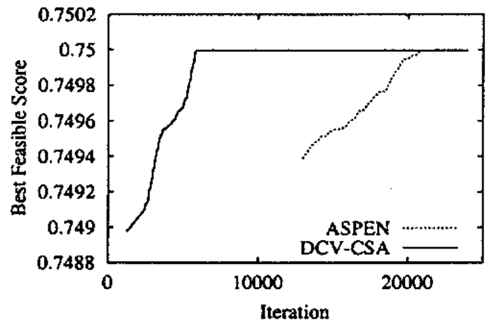
References

[1] E. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines*. J. Wiley and Sons, 1989.

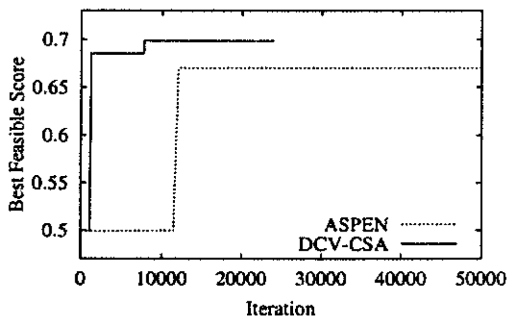
[2] J. A. Cadzow. Discrete calculus of variations. *Int. J. Control*, 11:393–407, 1970.

[3] S. E. Dreyfus. *Dynamic Programming and the Calculus of Variations*. Academic Press, New York and London, 1965.

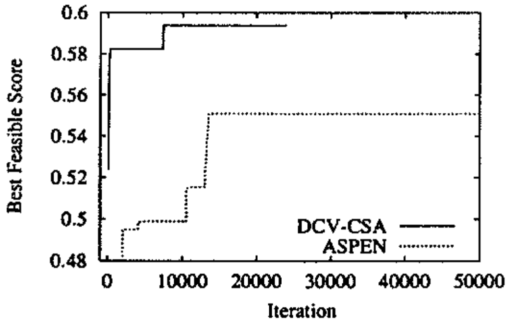
[4] S. Chien et al. Aspen - automating space mission operations using automated planning and scheduling. *SpaceOps*, 2000.



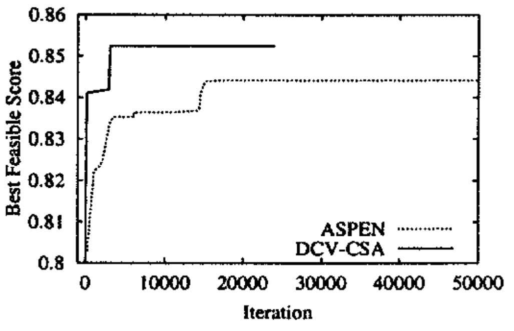
a) CX1-PREF benchmark with 8 orbits



b) CX1-PREF benchmark with 16 orbits



c) PREF benchmark



d) OPTIMIZE benchmark

Figure 4. Quality-time trade-offs in ASPEN and DCV-CSA on four benchmarks. (All DCV-CSA runs were terminated at 24,000 iterations.)

[5] A. K. Jónsson, P. H. Morris, N. Muscettola, and K. Rajan. Planning in interplanetary space: Theory and practice. *Proc. National Conference on Aritificial Intelligence*, 2000.

[6] B. W. Wah and Y. X. Chen. Chapter 10: Genetic algorithms for nonlinear constrained optimization. In X. Yao and R. Sarker, editors, *Evolutionary Optimization*, pages 253–275. Kluwer Academic Publishers, 2001.

[7] B. W. Wah and T. Wang. Simulated annealing with asymptotic convergence for nonlinear constrained global optimization. *Principles and Practice of Constraint Programming*, pages 461–475, October 1999.

[8] B. W. Wah and Z. Wu. The theory of discrete Lagrange multipliers for nonlinear discrete optimization. *Principles and Practice of Constraint Programming*, pages 28–42, October 1999.

[9] J. Willis, G. Rabideau, and C. Wilklow. The citizen explorer scheduling system. *Proceedings of the IEEE Aerospace Conference*, 1999.