Some Fixed-Point Results for the Dynamic Assignment Problem

Michael Z. Spivey
Department of Mathematics and Computer Science
Samford University, Birmingham, AL 35229

Warren B. Powell
Department of Operations Research and Financial Engineering
Princeton University, Princeton, NJ 08544

Corresponding Author: Michael Z. Spivey; Department of Mathematics and Computer Science, Samford University, Birmingham, AL 35299; *mzspivey@samford.edu*; 205-726-2454

Abstract

In previous work the authors consider the dynamic assignment problem, which involves solving sequences of assignment problems over time in the presence of uncertain information about the future. The algorithm proposed by the authors provides generally high-quality but non-optimal solutions. In this work, though, the authors prove that if the optimal solution to a dynamic assignment problem in one of two problem classes is unique, then the optimal solution is a fixed point under the algorithm.

Keywords: dynamic assignment problem, fixed point

A dynamic assignment problem consists of solving a sequence of assignment problems over time. At each time period decisions must be made as to which resources and tasks will or will not be assigned to each other. Assignments which are made at earlier time periods affect which assignments can be made during later time periods, and information about the future is often uncertain. Some examples of dynamic assignment problems include dispatching truck drivers to deliver loads, pairing locomotives with trains and assigning company employees to jobs.

In Spivey & Powell (2000) we define the dynamic assignment problem and establish several of its properties. We also develop a computationally tractable algorithm which provides high quality but generally not optimal solutions to the dynamic assignment problem. Our solution technique is based on solving the dynamic programming recursion

$$V_t(\mathcal{S}_t) = \max_{x_t \in \mathcal{X}_t} \; c_t \cdot x_t + V_{t+1}(\mathcal{S}_{t+1}),$$

where $V_t(\mathcal{S}_t)$ is the value of the system state $\mathcal{S}_t$ given time t, $\mathcal{X}_t$ is the available action space given time t, and c_t is the contribution function given time t. We avoid two of the "curses of dimensionality" (those of state and action spaces) by replacing $V_{t+1}(\mathcal{S}_{t+1})$ with a linear approximation $\hat{V}_{t+1}(\mathcal{S}_{t+1}) = v_{t+1} \circ \mathcal{S}_{t+1}$. This approach enables us to solve sequences of assignment problems, an algorithm which scales easily to large problems.

Testing our algorithm on deterministic instances of two dynamic assignment problem classes in which all resources are available initially produced unexpected results. In every instance the linear approximation based on the marginal values of the resources applied to the optimal solution produced the optimal solution in the subsequent iteration. In other words, if it was the case that $x_t^k = x_t^*$ for all t, where the vectors x_t^k, x_t^* are, respectively, the solution in iteration k, time t, and the optimal solution in time t, then we had

$$x_t^* = x_t^{k+1} = \text{argmax}_{x_t \in \mathcal{X}_t} \; c_t \cdot x_t + \hat{v}_{t+1}^k \circ \mathcal{S}_{t+1}.$$

Moreover, this implies that there exists a linear approximation $\hat{V}_t(\mathcal{S}_t)$ that produces the optimal solution.

The purpose of this paper is to prove this, provided the optimal solution is unique. Specifically, we show that:

- For deterministic instances of two dynamic assignment problem classes, if the optimal solution is unique then it is a fixed point under the algorithm presented in Spivey & Powell (2000).

In Section 1 we define some necessary notation and detail the problem classes for which the fixed-point results hold. In Section 2 we relate some structural properties of the static assignment problem and of deterministic dynamic assignment problems that we need to prove the fixed-point results. Section 3 outlines our solution strategy and algorithm based on approximating the value function using the values of individual resources. In Section 4 we present the experimental results that initially indicated that the optimal solution is a fixed point under our algorithm, as well as some other related results. The proofs of the fixed-point property for the two problem classes are contained in Section 5.

1 Notation and Problem Classes

We now give some necessary notation and define the problem classes we consider in this paper. Our notation is based on the model presented in Spivey & Powell (2000).

1.1 Resources and Tasks

We assume only finitely many resources and tasks available. Define:

$\mathcal{R} =$ Set of all possible resources.

$\mathcal{L} =$ Set of all possible tasks.

For a given time t, we can define:

$R_t =$ Vector of resources available given time t, where $R_{rt} = 1$ if resource r is available at t and 0 if not.

$L_t =$ Vector of tasks available given time t, where $L_{lt} = 1$ if task l is available at t and 0 if not.

We can also write the available resources and tasks given time t in set notation. These are induced from R_t and L_t in the following manner:

$\mathcal{R}_t = \{r \in \mathcal{R} \mid R_{rt} = 1\}.$

$$\mathcal{L}_t = \{l \in \mathcal{L} \mid L_{lt} = 1\}.$$

For the problem classes in this paper the form of randomness we consider is that of arrival times of tasks (we assume all resources are known initially).

Let Ω be a space of elementary outcomes, and define the following random variables:

$L_t^+ =$ vector of tasks that become known at time t; $L_{lt}^+ = 1$ if l becomes known at time t and 0 otherwise.

(In this paper R_t^+ is always the zero vector; thus we do not define it explicitly. However, we could have random arrivals of new resources as well as tasks.)

Finally, we define our *state variable*:

$$\mathcal{S}_t = (\mathcal{R}_t, \mathcal{L}_t).$$

1.2 Actions and System Dynamics

The actions given time t are the decisions to be made about assigning resources and tasks. Define:

$x_{rlt} =$ 1 if resource r is assigned to task l during time t and 0 otherwise.

$x_t =$ Vector consisting of $x_{rlt}\ \forall r \in \mathcal{R},\ \forall l \in \mathcal{L}$.

$R_t^A =$ Vector of resources assigned during time t, where $R_{rt}^A = \sum_{l \in \mathcal{L}} x_{rlt}$.

$L_t^A =$ Vector of tasks assigned during time t, where $L_{lt}^A = \sum_{r \in \mathcal{R}} x_{rlt}$.

At each time t we maintain knowledge of the resource and task vectors R_t and L_t.We use these to specify the system dynamics:

$$(1) \quad R_{t+1} = R_t - R_t^A$$

$$(2) \quad L_{t+1} = L_t + L_{t+1}^+ - L_t^A$$

$$(3) \quad \mathcal{S}_{t+1} = (\mathcal{R}_{t+1}, \mathcal{L}_{t+1})$$

The assignments that can be made during time t are constrained by the availability of the resources and tasks given t. Specifically, the feasible action set $\mathcal{X}_t(\mathcal{S}_t)$ is the set of all assignment

vectors x_t that satisfy the following constraints:

$$\begin{aligned}
\sum_{l \in \mathcal{L}} x_{rlt} &\leq R_{rt} \;\; \forall r \in \mathcal{R}, \\
\sum_{r \in \mathcal{R}} x_{rlt} &\leq L_{lt} \;\; \forall l \in \mathcal{L}, \\
x_{rlt} &\in \{0,1\} \;\; \forall r \in \mathcal{R}, \; \forall l \in \mathcal{L}.
\end{aligned}$$

Later we develop specific techniques for making assignments. For now, define a *policy*:

$$\pi : \mathcal{S} \rightarrow x.$$

A policy is simply a function that maps states to decisions. We use policies to determine the action to take when a system is in state $\mathcal{S}$.

1.3 Objective Function

Define

$c_{rlt} =$ contribution for assigning resource r to task l given time t.

Then the contribution for taking action x_t given time t is:

$$c_t \cdot x_t = \sum_{r \in \mathcal{R}} \sum_{l \in \mathcal{L}} c_{rlt} \, x_{rlt}.$$

For a state $\mathcal{S}_t$ and a policy π, define, for each t:

$$F_t^{\pi}(\mathcal{S}_t) = E\Big[\sum_{t'=t}^{T+1} c_{t'} \cdot x_{t'}^{\pi} \Big| \mathcal{S}_t \Big].$$

And define:

$$F_t^*(\mathcal{S}_t) = \max_{\pi} \; F_t^{\pi}(\mathcal{S}_t).$$

The solution to our dynamic assignment problem can be found by solving:

$$F_0^*(\mathcal{S}_0) = \max_{\pi} F_0^{\pi}(\mathcal{S}_0).$$

1.4 Problem Classes

Spivey & Powell (2000) consider general dynamic assignment problems with the following characteristics:

- Once assigned, resources and tasks disappear from the system.
- Once a resource/task pair becomes known, it is available for assignment, and the upper bound and contribution for that pair are known immediately.
- The contribution for a resource/task pair is a concave and monotonically decreasing function of time from the point at which it becomes known until it reaches 0.

We consider the following two special cases:

Problem Class 1:

- All resources are available initially. Tasks arrive over time, with at most one task arriving at any given time.
- Upon arrival, resources and tasks remain in the system until assigned.

Problem Class 2: Same as Problem Class 1 except that tasks disappear from the system if not assigned immediately.

2 Some Structural Properties

To establish the fixed-point results we need some structural properties of the static assignment problem and of deterministic dynamic assignment problems, all of which are either established or cited in Spivey & Powell (2000). Therefore we state them here without proof.

2.1 Some Properties of the Static Assignment Problem

Given the direct product of sets $\mathcal{S}' = (\mathcal{R}', \mathcal{L}')$ consisting of resources and tasks to be assigned and contributions c_{rl} for assigning each resource r to each task l, define:

$$C(\mathcal{S}') = \quad \max_x \quad c \cdot x$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{l \in \mathcal{L}} x_{rl} \leq 1 \ \ \forall r \in \mathcal{R}', \\
& \sum_{r \in \mathcal{R}} x_{rl} \leq 1 \ \ \forall l \in \mathcal{L}', \\
& x_{rl} \in \{0,1\} \ \ \forall r \in \mathcal{R}, \ \forall l \in \mathcal{L}, \\
& x_{rl} = 0 \quad \text{if } r \notin \mathcal{R}' \text{ or } l \notin \mathcal{L}'.
\end{aligned}$$

Given a network $\mathcal{S}' = (\mathcal{R}', \mathcal{L}')$, define:

$X^*(\mathcal{S}') =$ the set of optimal assignments for $\mathcal{S}'$.

$x^*(\mathcal{S}') =$ an element of $X^*(\mathcal{S}')$.

$l^*(r) =$ the task assigned to resource r under $x^*(\mathcal{S}')$. If r is not assigned under x^*, then $l^*(r)$ is the supersink.

$r^*(l) =$ the resource assigned to task l under $x^*(\mathcal{S}')$. If l is not assigned under x^*, then $r^*(l)$ is the supersource.

$C^{r+}(\mathcal{S}') = C(\mathcal{S}' \cup \{r\})$, for $r \notin \mathcal{R}'$.

$C^{r-}(\mathcal{S}') = C(\mathcal{S}' - \{r\})$, for $r \in \mathcal{R}'$.

$C^{rl+}(\mathcal{S}') = C(\mathcal{S}' \cup \{r\} \cup \{l\})$, for $r \notin \mathcal{R}', l \notin \mathcal{L}'$.

$C^{r_1 r_2 +}(\mathcal{S}') = C(\mathcal{S}' \cup \{r_1, r_2\})$, for $r_1, r_2 \notin \mathcal{R}'$.

$C^{l+}(\mathcal{S}'), C^{l-}(\mathcal{S}'), C^{rl-}(\mathcal{S}')$ and $C^{r_1 r_2 -}(\mathcal{S}')$ are defined similarly.

Our proofs rely on the value of adding or removing resources and tasks from a system. A fundamental result relating these is due to Shapley (1962):

Theorem 1 (Shapley 1962) *Given a network $\mathcal{S}'$,*

1. $\Big(C(\mathcal{S}' \cup \{r_1\}) - C(\mathcal{S}')\Big) + \Big(C(\mathcal{S}' \cup \{r_2\}) - C(\mathcal{S}')\Big) \geq C(\mathcal{S}' \cup \{r_1, r_2\}) - C(\mathcal{S}')$.
2. $\Big(C(\mathcal{S}' \cup \{l_1\}) - C(\mathcal{S}')\Big) + \Big(C(\mathcal{S}' \cup \{l_2\}) - C(\mathcal{S}')\Big) \geq C(\mathcal{S}' \cup \{l_1, l_2\}) - C(\mathcal{S}')$.
3. $\Big(C(\mathcal{S}' \cup \{r\}) - C(\mathcal{S}')\Big) + \Big(C(\mathcal{S}' \cup \{l\}) - C(\mathcal{S}')\Big) \leq C(\mathcal{S}' \cup \{r, l\}) - C(\mathcal{S}')$.

Our proofs also rely on the concept of flow-augmenting paths (see, for instance, Powell (1989)). Define the contribution of a flow-augmenting path y^{mn} to be

$$C(y^{mn}) = c \cdot y^{mn}.$$

And define

$$y^{mn^*} = \arg\max_{y^{mn}} C(y^{mn}).$$

Powell (1989) proves that the flow-augmenting paths resulting from adding resources and removing tasks (or vice versa) from the system form a flow-augmenting tree in the network. We also require the following result:

Theorem 2 (Powell 1989) *If SSE is the supersource, SSK the supersink, r a resource and l a task, we have (assuming the resulting augmented networks are feasible):*

1. $C^{r+}(\mathcal{S}') - C(\mathcal{S}') = C(y^{r,SSK^*})$
2. $C^{r-}(\mathcal{S}') - C(\mathcal{S}') = C(y^{SSK,r^*})$
3. $C^{l+}(\mathcal{S}') - C(\mathcal{S}') = C(y^{SSE,l^*})$
4. $C^{l-}(\mathcal{S}') - C(\mathcal{S}') = C(y^{l,SSE^*})$
5. $C^{rl+}(\mathcal{S}') - C(\mathcal{S}') = C(y^{r,l^*})$
6. $C^{rl-}(\mathcal{S}') - C(\mathcal{S}') = C(y^{l,r^*})$

From this we have a few corollaries Spivey & Powell (2000):

Corollary 1 *Given a network $\mathcal{S}'$, if r is not assigned under the optimal solution x^*, then $C(\mathcal{S}') = C^{r-}(\mathcal{S}')$.*

Corollary 2 *Given a network $\mathcal{S}'$, if r is not assigned under the optimal solution x^*, then, for any $l \in \mathcal{S}'$, $C^{l-}(\mathcal{S}') = C^{rl-}(\mathcal{S}')$.*

Corollary 3 *Given a network $\mathcal{S}'$, if r is not assigned under the optimal solution x^*, then, for any $l \in \mathcal{S}'$, there exists a maximal flow-augmenting path from l to SSE that does not include r.*

The versions of Corollaries 1, 2 and 3 for unassigned tasks are also true.

For a network $\mathcal{S}'$, we define:

$$v^{r+} = C^{r+}(\mathcal{S}') - C(\mathcal{S}').$$

$$v^{r-} = C(\mathcal{S}') - C^{r-}(\mathcal{S}') \text{ (provided } r \in \mathcal{R}').$$

Define v^{l+}, v^{l-} similarly. By definition, then, we have

$$v^{r+}(\mathcal{S}') = v^{r-}(\mathcal{S}' \cup \{r\}).$$

We also define:

$$v^{rl+} = C^{rl+}(\mathcal{S}') - C(\mathcal{S}').$$

$$v^{rl-} = C(\mathcal{S}') - C^{rl-}(\mathcal{S}') \text{ (provided } r \in \mathcal{R}', l \in \mathcal{L}').$$

By definition we also have $v^{rl+}(\mathcal{S}) = v^{rl-}(\mathcal{S} \cup \{r\} \cup \{l\})$.

Then we have:

Corollary 4 *Given a network $\mathcal{S}'$, the following are true:*

1. *If r (l) is not assigned under an optimal solution x^*, then $v^{r-}(v^{l-}) = 0$.*
2. *If r and l are assigned under an optimal solution x^*, then $v^{rl-} = c_{rl}$.*

2.2 Some Properties of Deterministic Dynamic Assignment Problems

To establish the fixed-point results we also need the following properties of deterministic dynamic assignment problems, both of which are established in Spivey & Powell (2000).

Proposition 1 *Consider a deterministic dynamic assignment problem in which c_{rlt} is a strictly decreasing function of t. Let τ^a_{rl} be the first time resource r and task l are both available. If $x^*_{rlt} = 1$ for some t in an optimal solution x^*, then $x^*_{rl\tau^a_{rl}} = 1$.*

Given a deterministic assignment problem, let $\bar{\mathcal{S}}_t = (\bar{\mathcal{R}}_t, \bar{\mathcal{L}}_t)$ be the underlying network given time t under the optimal solution x^*; i.e., $\bar{\mathcal{R}}_t = \{r \in \mathcal{R} : x^*_{rlt'} = 0 \ \forall l, \forall t' < t\}, \bar{\mathcal{L}}_t = \{l \in \mathcal{L} : x^*_{rlt'} = 0 \ \forall r, \forall t' < t\}$ and $c_{rl} = c_{r,l,\max\{\tau^a_{rl},t\}} \forall r \in \bar{\mathcal{R}}_t, \forall l \in \bar{\mathcal{L}}_t$. Let $\bar{\mathcal{R}}_t{}^- = \bar{\mathcal{R}}_t - \{r : x^*_{rlt} = 1 \text{ for some } l \in \bar{\mathcal{L}}_t\}$ and $\bar{\mathcal{L}}_t{}^- = \bar{\mathcal{L}}_t - \{l : x^*_{rlt} = 1 \text{ for some } r \in \bar{\mathcal{R}}_t\}$. Then let $\bar{\mathcal{S}}_t{}^- = (\bar{\mathcal{R}}_t{}^-, \bar{\mathcal{L}}_t{}^-)$. Both $\bar{\mathcal{S}}_t$ and $\bar{\mathcal{S}}_t{}^-$ are subnetworks of the equivalent network formulation $\mathcal{S}'$.

Then we have:

Corollary 5 $C(\bar{\mathcal{S}}_t^{-}) = C(\bar{\mathcal{S}}_{t+1})$.

3 Solution Strategy and Algorithm

We define the value function as follows:

$$
\begin{aligned}
(4) \quad V_t(\mathcal{S}_t) &= \max_{x_t \in \mathcal{X}_t} \Big\{ c_t \cdot x_t + E[V_{t+1}(\mathcal{S}_{t+1}) | \mathcal{S}_t] \Big\}; \;\; t = 0, \ldots, T, \\
&= 0; \;\; t = T+1.
\end{aligned}
$$

Our solution strategy is based on approximating the value function $V_t(\mathcal{S}_t)$. In Spivey & Powell (2000) we consider three main classes of approximations: 1) the greedy or myopic approximation, 2) separable linear approximations and 3) two-index nonseparable approximations. The second and third classes are based on approximating the values of resources and tasks: The second considers individual resources and tasks while the third focuses on the values of resource/task pairs. It is the approximation in the second class which uses the values of individual resources that gives us the fixed-point results in this paper.

We consider the value of a resource to be the change in the total system contribution resulting from adding or removing the resource from the system. As this is a marginal effect of the resource on the system contribution we refer to the value as a resource *gradient*.

$$
v_{rt}(\mathcal{S}_t) = \begin{cases} V_t(\mathcal{S}_t) - V_t(\mathcal{S}_t - \{r\}) & \text{if } r \in \mathcal{S}_t, \\ V_t(\mathcal{S}_t \cup \{r\}) - V_t(\mathcal{S}_t) & \text{if } r \notin \mathcal{S}_t. \end{cases}
$$

The value or gradient of a task is defined correspondingly.

We can approximate $V_t(\mathcal{S}_t)$ with the resource and task gradients in the following ways:

$$
\begin{aligned}
(5) \quad \hat{V}_t^r(\mathcal{S}_t) &= \sum_{r \in \mathcal{R}_t} v_{rt}, \\
(6) \quad \hat{V}_t^l(\mathcal{S}_t) &= \sum_{l \in \mathcal{L}_t} v_{lt}, \\
(7) \quad \hat{V}_t^{r,l}(\mathcal{S}_t) &= \sum_{r \in \mathcal{R}_t} v_{rt} + \sum_{l \in \mathcal{L}_t} v_{lt}.
\end{aligned}
$$

Approximating the value function (4) requires approximating $E[V_{t+1}(\mathcal{S}_{t+1})|\mathcal{S}_t]$. Using the resource gradients approximation $\hat{V}^r_{t+1}(\mathcal{S}_{t+1})$ for $E[V_{t+1}(\mathcal{S}_{t+1})|\mathcal{S}_t]$ yields (see Spivey & Powell (2000) for details):

$$(8) \quad \hat{V}^r_t(\mathcal{S}_t) \;=\; \max_{x_t \in X_t} \left\{ \sum_{r\in\mathcal{R}} \sum_{l\in\mathcal{L}} (c_{rlt} - v_{r,t+1}) \cdot x_{rlt} \right\}.$$

We now present our algorithm. The algorithm is based on our linear approximation (8) of $V_t(\mathcal{S}_t)$. For each iteration k and time t we formulate and solve a network problem consisting of the available resources and tasks, the assignment contributions, the upper bounds on the assignment arcs and the resource gradients from the previous iteration. The resource gradients take the form of contributions on the arcs from the resources to the supersink; in effect, they are contributions for not assigning the resources. After solving the network problem we remove the assigned resources and tasks from the system and proceed to the next time period (or iteration, if we are in the final time period).

We represent the resource gradients in the following manner:

$\hat{v}^k_{rt} =$ the gradient approximation of resource r directly obtained during iteration k, time t, and

$\bar{v}^k_{rt} =$ the smoothed gradient approximation of resource r given time t after iteration k. In particular, for smoothing function α^k, $\bar{v}^k_{rt} = \alpha^k\, \hat{v}^k_{rt} + (1-\alpha^k)\, \bar{v}^{k-1}_{rt}$.

Our algorithm is as follows:

Step 0. Determine a maximum number of iterations K. Set $\hat{v}^0_{rt} = 0$ and $\bar{v}^0_{rt} = 0$ for all r and t. Set $k = 1, t = 0$.

Step 1. For the current k and t, solve the assignment problem

$$(9) \quad \begin{aligned} \hat{V}^{r,k}_t(\mathcal{S}_t) = \quad & \max_{x_t} \quad \left\{ \sum_{r\in\mathcal{R}} \sum_{l\in\mathcal{L}} (c_{rlt} - \bar{v}^{k-1}_{r,t+1}) \cdot x_{rlt} \right\} \\ & \text{subject to} \quad \sum_{l\in\mathcal{L}} x_{rlt} \le R_{rt} \;\; \forall r \in \mathcal{R}, \\ & \qquad\qquad \sum_{r\in\mathcal{R}} x_{rlt} \le L_{lt} \;\; \forall l \in \mathcal{L}, \\ & \qquad\qquad x_{rlt} \in \{0,1\} \;\; \forall r \in \mathcal{R},\; \forall l \in \mathcal{L}. \end{aligned}$$

Step 2. (Transition.) Once the argmax x_t in Step 1 is determined, let $R_{t+1} = R_t - R_t^A$. For Problem Class 1, $L_{t+1} = L_t + L_{t+1}^+ - L_t^A$, and for Problem Class 2, $L_{t+1} = L_{t+1}^+$, as tasks are not persistent.

Step 3. If $t < T$ then $t = t + 1$ and go to Step 2.

Step 4. (Backwards calculation of resource gradients.) Let N be the network consisting of all resources and tasks available at iteration k and times $t' \geq t$. Let $c_{rl} = c_{r,l,\tau_{rl}^a}$. Then, for the current k and t, and for each r and l that become available by time t (even if one or both were assigned before time t), calculate $\hat{v}_{rt}^k$ according to one of the following cases:

1. If r is available given time t, then $\hat{v}_{rt}^k = C(N) - C^{r-}(N)$.
2. If r is not available given time t, then $\hat{v}_{rt}^k = C^{r+}(N) - C(N)$.

Step 5. (Smoothing.) For each r, set $\bar{v}_{rt}^k = \alpha^k \, \hat{v}_{rt}^k + (1 - \alpha^k) \, \bar{v}_{rt}^{k-1}$ (for some smoothing function α^k).

Step 6. If $t > 0$ then $t = t - 1$ and go to Step 4.

Step 7. If $k < K$ then $k = k + 1$ and go to Step 1 else stop.

The gradients $\hat{v}_{rt}^k$ for iteration k can be calculated in one of two ways: 1) with flow-augmenting paths, or 2) numerically. The numerical calculations involve calculating both $C(N)$ and $C^{r-}(N)$ (or $C^{r+}(N)$ and $C(N)$, if r is not available) and taking the difference. These calculations can be quite fast if one uses the solution of $C(N)$ as a warm start when calculating $C^{r-}(N)$ and $C^{r+}(N)$.

The algorithm has two stages in each iteration: a forward pass and a backward pass. In the forward pass the stochastic process is realized and assignments are made. The gradients are calculated in the backward pass. As the gradients are not calculated until after the entire stochastic process is realized we can use the information from times later than t when calculating the gradients during time t. Since this calculation involves more information than just that available given time t we believe that we are more accurately capturing the values of the gradients than if we were to calculate them during the the initial forward pass. This does not violate non-anticipativity restrictions because the gradients from iteration k are not used until iteration $k + 1$, when all information from iteration k is known anyway.

Another issue is that the backwards calculation of gradients requires an underlying basis solution

from which to determine which resources are available at each time period. The most straightforward basis solution to use is the one from the current iteration; i.e., to calculate the gradients for iteration k we use the solution from iteration k. However, at the time the gradients are calculated all of the information from iteration k, including the history of the entire stochastic process, is known. Thus we do not have to use the solution from iteration k as our basis solution; we could even use the post-optimal solution as the basis.

There are three basic versions of the algorithm tested in Spivey & Powell (2000): with resource gradients only, with resource and task gradients and with arc gradients. Of these three, the resource gradients version gives the worst results on the DAP class discussed in that paper. However, for the problem classes under discussion in this paper using the resource gradients is actually sufficient to produce the fixed-point results. This could be because the purpose of a gradient at a particular point in time is to decide whether or not to hold a resource and/or task over to the next time period, with the hope of assigning the resource and/or task for a higher contribution at some later point. In both problem classes under discussion all resources are available at time 0 and contributions decrease monotonically. For the class in which tasks are persistent, it is thus the case that holding a task l at time t will never result in l being assigned at a time later than t to a resource for a contribution higher than one at t. For the class in which tasks are not persistent, it makes no sense to hold tasks for a higher contribution later because they disappear if not immediately assigned. Thus it seems intuitive that for these problem classes using resource gradients only is sufficient.

4 Experimental Results

To test our algorithm on Problem Classes 1 and 2 we developed a basic set of twenty problems. We represented resources and tasks as random points on a grid and defined the contribution for assigning a resource and a task to be an inverse function of the distance between their corresponding grid points.

In presenting our results we refer to an "optimal solution" and a "myopic solution." "Optimal solution" is the posterior optimal solution. "Myopic solution" is that found by solving the assignment problem at each time period using only the resources and tasks available given that time and with $\bar{v} = 0$.

Tables 1 and 2 compare the use of the resource gradients algorithm on deterministic instances

of the two problem classes: The first gives the results for Problem Class 1, in which resources are persistent, and the second the results for Problem Class 2, in which resources are not persistent. The first column in each table is a measure of the size of the data set. The second and third columns give the results after one iteration using the post-optimal and myopic solutions, respectively, as the basis solutions from which to calculate the gradients. The fourth column shows the solution converged to by the algorithm using the myopic solution as the basis solution in the first iteration and the current solution as the basis solution in each subsequent iteration, with a step size of 0.05. All numbers are given as a percent of the post-optimal solution.

The results in column two were a huge surprise. We did not expect that using the optimal solution as the basis from which to calculate the gradients would produce the optimal solution in the subsequent iteration, much less on every data set. These results suggested the fixed-point property that we were eventually able to prove and that we present in Section 5.

We also wished to determine if producing the optimal solution in one iteration is a function of using the optimal as the basis solution or simply of using resource gradients. We ran more tests using the myopic solution as the basis. Column three in Tables 1 and 2 presents these results. We can see that using the myopic solution as the basis does not, in general, produce the optimal solution in the subsequent iteration. Thus achieving the optimal solution in one iteration is not simply a function of using resource gradients.

Column four shows the convergence results using the myopic solution as the basis solution in the first iteration and the current solution as the basis solution on subsequent iterations. The purpose of this experiment was twofold: to test the convergence properties of the algorithm and to see if the myopic solution is, like the optimal, a fixed-point. The results are virtually identical for the two problem classes. They are also near-optimal for most data sets, with a mean of 99.2 and a median of 99.7. Only on four data sets in each problem class did the algorithm converge to a solution less than 99% of optimal. Moreover, in most cases the results in columns three and four are different. Thus the myopic solution is not, in general, a fixed point, unlike the optimal solution.

5 Fixed-Point Proofs

We now prove the following:

Theorem 3 *For a deterministic dynamic assignment problem in Problem Class 1, if the optimal solution is unique then it is a fixed point under our algorithm.*

Proof: Let x^* be the optimal solution, and let x^{OPT} be the solution produced by the gradients derived from the optimal solution. We wish to show that $x^{OPT} = x^*$. The proof will be by induction. The base case $t = 0$ does not need to be dealt with explicitly because when $t = 0$ the assumption in the induction step is true by default; thus the proof in the base case is identical to the proof in the induction step.

Given time t there are five parts of the proof. We need to show that, under x^*:

1. If task l is not assigned during time t, then
   (a) $\hat{v}_{r,t+1} > c_{rlt}$ for all r available and unassigned during t, and
   (b) if r' is assigned during t to l', then
      i. $c_{r'lt} < c_{r'l't}$, and
      ii. for any r'' available during t, $c_{r''l't} + c_{r'lt} < c_{r'l't} + \hat{v}_{r'',t+1}$.
2. If task l is assigned during time t to r, then
   (a) $\hat{v}_{r,t+1} < c_{rlt}$, and
   (b) for any r' available and unassigned during t, $c_{r'lt} + \hat{v}_{r,t+1} < c_{rlt} + \hat{v}_{r',t+1}$.

Since all resources are available initially and at most one task l' becomes available during t, by Proposition 1 there can be at most one assignment under the optimal solution; namely, l' possibly being assigned to some r. The items enumerated above cover the cases:

1(a). If l is not assigned during time t under x^*, then no resource unassigned during t under x^* is assigned to l during t under x^{OPT}.

1(b)i. If l is not assigned and r' is assigned to l' during time t under x^*, then it is better to assign r' to l' than to l during t (i.e., r' will not be assigned to l during t) under x^{OPT}.

1(b)ii. If r'' and l are not assigned and r' is assigned to l' during time t under x^*, then it is better to assign r' to l' and hold r'' and l than to assign r' to l and r'' to l' during t (i.e., r' will not be assigned to l and r'' to l' during t) under x^{OPT}.

Thus, in Case 1, if l is not assigned during t under x^* then it will not be assigned during t under x^{OPT}.

2(a). If r and l are assigned to each other during time t under x^*, then it is better to assign r to l than to hold r and l during t (i.e., r and l will not both be held during t) under x^{OPT}.

2(b). If r and l are assigned to each other and r' is not assigned during time t under x^*, then it is better to assign r to l and hold r' than to assign r' to l and hold r during t (ie., l will not be assigned to any resource other than r during t) under x^{OPT}.

Therefore, in Case 2, if l is assigned to r during time t under x^* then l will be also be assigned to r under x^{OPT}.

These cover all cases.

Given time t, suppose that $x^{OPT}_{t'} = x^*_{t'}$ for all $t' < t$. Then, since $\mathcal{R}^{OPT}_0 = \mathcal{R}$, by the induction step we have $\mathcal{R}^{OPT}_t = \{r \in \mathcal{R} : x^*_{rlt'} = 0 \ \forall l, \forall t' < t\}$. Also by the induction step we have $\mathcal{L}^{OPT}_t = \{l \in \mathcal{L} : \exists\, t' < t \ni l \in \mathcal{L}^+_{t'} \cup \mathcal{L}^{OPT}_0, x^*_{rlt} = 0 \ \forall r \, \forall t\} \cup \mathcal{L}^+_t$. We wish to show that $x^{OPT}_t = x^*_t$.

Case 1. Let $\bar{\mathcal{S}}_t{}^-, \bar{\mathcal{S}}_{t+1}$ be as in Corollary 5. Let $l \in \mathcal{L}^{OPT}_t$ and suppose $x^*_{rlt} = 0 \ \forall r$. Then $l \in \mathcal{L}^*_{t+1}$, and, by Proposition 1, l is not assigned under x^*.

Part a. Let $r \in \mathcal{R}^{OPT}_t$ and suppose $r \in \mathcal{R}^*_{t+1}$. Then $r, l \in \bar{\mathcal{S}}_t{}^-, \bar{\mathcal{S}}_{t+1}$. Let $\bar{x_t}^{rl-}$ be the optimal assignment vector under $\bar{\mathcal{S}}_t{}^- - \{r, l\}$. Then $\bar{x_t}^{rl-}$ is feasible for $\bar{\mathcal{S}}_t{}^-$, as $\bar{\mathcal{S}}_t{}^- - \{r, l\} \subset \bar{\mathcal{S}}_t{}^-$. Moreover, $\bar{x_t}^{rl-} + 1_{rl}$ (where 1_{rl} is a vector of zeros with a one in element (r, l)) is feasible for $\bar{\mathcal{S}}_t{}^-$. Thus by uniqueness we have

$$\begin{aligned} & c_t \cdot (x^{rl-}_t + 1_{rl}) < C(\bar{\mathcal{S}}_t{}^-) \\ \Rightarrow \quad & c_t \cdot x^{rl-}_t + c_{rlt} < C(\bar{\mathcal{S}}_t{}^-) \\ \Rightarrow \quad & C^{rl-}(\bar{\mathcal{S}}_t{}^-) + c_{rlt} < C(\bar{\mathcal{S}}_t{}^-). \end{aligned}$$

Since $\bar{\mathcal{R}}_t{}^- = \bar{\mathcal{R}}_{t+1}$ and $\bar{\mathcal{L}}_t{}^- = \bar{\mathcal{L}}_{t+1}$, the set of feasible assignments for $\bar{\mathcal{S}}_t{}^-$ and $\bar{\mathcal{S}}_{t+1}$ are the same. Because $c_{r,l,t+1} \leq c_{rlt} \ \forall r \in \bar{\mathcal{R}}_{t+1}, \ \forall l \in \bar{\mathcal{L}}_{t+1}$, for any feasible assignment x_{t+1} we have $c_{t+1} \cdot x_{t+1} \leq c_t \cdot x_{t+1}$. Removal of the same resources and tasks from $\bar{\mathcal{S}}_t{}^-$ and $\bar{\mathcal{S}}_{t+1}$ results in two networks with the same two properties; e.g., $\bar{\mathcal{S}}_t{}^- - \{r, l\}$ and $\bar{\mathcal{S}}_{t+1} - \{r, l\}$. In particular, then, $C^{rl-}(\bar{\mathcal{S}}_{t+1}) \leq C^{rl-}(\bar{\mathcal{S}}^-_t)$ for any $r \in \bar{\mathcal{R}}_{t+1}, l \in \bar{\mathcal{L}}_{t+1}$. By Corollary 5 we also have $C(\bar{\mathcal{S}}_t{}^-) = C(\bar{\mathcal{S}}_{t+1})$. Thus the above inequality gives us $C^{rl-}(\bar{\mathcal{S}}_{t+1}) + c_{rlt} < C(\bar{\mathcal{S}}_{t+1})$. Since l is not assigned under x^*, by Corollary 2 we have $C^{rl-}(\bar{\mathcal{S}}_{t+1}) = C^{r-}(\bar{\mathcal{S}}_{t+1})$. Thus $C^{r-}(\bar{\mathcal{S}}_{t+1}) + c_{rlt} < C(\bar{\mathcal{S}}_{t+1})$. Since $\hat{v}_{r,t+1} = C(\bar{\mathcal{S}}_{t+1}) - C^{r-}(\bar{\mathcal{S}}_{t+1})$, we have $c_{rlt} < \hat{v}_{r,t+1}$.

Part bi. Suppose $r' \in \mathcal{R}^{OPT}_t$ is assigned to some l' during time t under x^*. Let x' be a solution

to the problem such that $x'_{rlt'} = x^*_{rlt'} \ \ \forall r \, \forall l \, \forall t'$ except that $x'_{r'lt} = 1$ and $x'_{r'l't} = 0$ where $x^*_{r'lt} = 0$ and $x^*_{r'l't} = 1$. By uniqueness, $c \cdot x^* > c \cdot x'$, which implies $c_{r'lt} \cdot x^*_{r'lt} + c_{r'l't} \cdot x^*_{r'l't} > c_{r'lt} \cdot x'_{r'lt} + c_{r'l't} \cdot x'_{r'l't}$. Thus $c_{r'l't} > c_{r'lt}$.

Part bii. Let $r', r'' \in \mathcal{R}^{OPT}_t$. Suppose r' is assigned to some l' during time t under x^* and $r'' \in \mathcal{R}^*_{t+1}$. First, we show that $\hat{v}_{r'',t+1} = v_{r''}(\bar{\mathcal{S}}_t{}^-)$. By the definition of the transition function, $\bar{\mathcal{S}}_{t+1} = \bar{\mathcal{S}}_t{}^-$ except that contributions between some resources and tasks are higher in $\bar{\mathcal{S}}_t{}^-$. In particular, the contributions that change are exactly those between resources and tasks that are available both during time t and time $t+1$ under x^*. By Proposition 1, the only tasks available during times t and $t+1$ are those that are never assigned under x^*. By repeated application of Corollary 2 (effectively, removing all tasks that are never assigned) we thus have that $C^{r''-}(\bar{\mathcal{S}}_t{}^-) = C^{r''-}(\bar{\mathcal{S}}_{t+1})$. By Corollary 5 we also have $C(\bar{\mathcal{S}}_t{}^-) = C(\bar{\mathcal{S}}_{t+1})$. Therefore,

$$\begin{aligned} v_{r''}(\bar{\mathcal{S}}_t{}^-) &= C(\bar{\mathcal{S}}_t{}^-) - C^{r''-}(\bar{\mathcal{S}}_t{}^-) \\ &= C(\bar{\mathcal{S}}_{t+1}) - C^{r''-}(\bar{\mathcal{S}}_{t+1}) \\ &= \hat{v}_{r'',t+1}. \end{aligned}$$

Now we show that $c_{r'lt} + c_{r''l't} - v_{r''}(\bar{\mathcal{S}}_t{}^-) < c_{r'l't}$. By Corollary 3 there exists a maximal flow-augmenting path $y^{SSK,r''*}$ in $\bar{\mathcal{S}}_t{}^-$ that does not include l. Moreover, $y^{SSK,r''*}$ does not include l' (as $l' \notin \bar{\mathcal{S}}_t{}^-$). Therefore the paths $1_{r'lt}$ and $y^{SSK,r''*} + 1_{r''l't}$ are disjoint. Thus we have a feasible flow-augmenting path from r' to l' in $\bar{\mathcal{S}}_t{}^-$ consisting of the arcs (r', l), (l, SSE), (SSE, SSK), the arcs defined by $y^{SSK,r''*}$, and (r'', l'). This path defines a feasible solution $x^*(\bar{\mathcal{S}}_t{}^-) + 1_{r'l} + 1_{l,SSE} + 1_{SSE,SSK} + y^{SSK,r''*} + 1_{r''l'}$ to $\bar{\mathcal{S}}_t$. Since $x^*(\bar{\mathcal{S}}_t)$ is unique, we have $c_t \cdot (x^*(\bar{\mathcal{S}}_t{}^-) + 1_{r'l} + 1_{l,SSE} + 1_{SSE,SSK} + y^{SSK,r''*} + 1_{r''l'}) < c_t \cdot x^*(\bar{\mathcal{S}}_t)$, provided the two solutions are distinct. Since $x_{r'l't} = 1$ in $x^*(\bar{\mathcal{S}}_t)$ but not in the other solution, they are distinct. By Corollary 4 we have $C(\bar{\mathcal{S}}_t) = C(\bar{\mathcal{S}}_t{}^-) + c_{r'l't}$. Thus $c_{r'l't} > c_{r'lt} + C(y^{SSK,r''*}) + c_{r''l't}$, which implies $c_{r'l't} > c_{r'lt} - v_{r''}(\bar{\mathcal{S}}_t{}^-) + c_{r''l't}$.

Since $\hat{v}_{r'',t+1} = v_{r''}(\bar{\mathcal{S}}_t{}^-)$, we therefore have $c_{r'l't} + \hat{v}_{r'',t+1} > c_{r'lt} + c_{r''l't}$.

Case 2. Let $l \in \mathcal{L}^{OPT}_t$, $r \in \mathcal{R}^{OPT}_t$, and suppose $x^*_{rlt} = 1$.

Part a. First, we show that $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$. Let y^{r,SSK^*} be a maximal flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_{t+1}$. Since, by the proof of Corollary 5, $\bar{\mathcal{S}}_t{}^-$ and $\bar{\mathcal{S}}_{t+1}$ have the same optimal assignments, y^{r,SSK^*} is a feasible flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t{}^-$. Either y^{r,SSK^*} contains a task unassigned in $\bar{\mathcal{S}}_{t+1}$ or not. If not, then by an argument similar to that in Case 1bii the contribution for each link in y^{r,SSK^*} in the network $\bar{\mathcal{S}}_{t+1}$ is the same as the contribution

for the associated link in $\bar{\mathcal{S}}_t{}^-$. Since y^{r,SSK^*} is a feasible flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t{}^-$, we have $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$. If y^{r,SSK^*} contains a task l' unassigned in $\bar{\mathcal{S}}_{t+1}$, then in order to be feasible y^{r,SSK^*} must contain the mirror link (l', SSE) and a link (r', l') for some resource r'. Moreover, we can now see that y^{r,SSK^*} can contain at most one such unassigned task, as otherwise the supersource would occur twice in the path, creating a cycle and contradicting the tree of flow-augmenting paths described in Powell (1989). Thus y^{r,SSK^*} consists of a path from r to r' containing no unassigned tasks, the link (r', l'), the mirror link (l', SSE) and a path from SSE to SSK. We have already seen that the path from r to r' and the path from SSE to SSK contain no tasks unassigned in $\bar{\mathcal{S}}_{t+1}$; thus as in an earlier argument the contribution for each link in these paths is the same as that for the associated link in $\bar{\mathcal{S}}_t{}^-$. Since $c_{r'l',t+1} \leq c_{r'l't}$ we have that the length of y^{r,SSK^*} in $\bar{\mathcal{S}}_t{}^-$ is greater than or equal to its length in $\bar{\mathcal{S}}_{t+1}$. And because y^{r,SSK^*} is feasible for $\bar{\mathcal{S}}_t{}^-$ this gives us $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$. In either case, then, $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$.

Now we show that $v_r(\bar{\mathcal{S}}_t{}^-) < c_{rlt}$. Let y^{r,SSK^*} be a maximal flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t{}^-$. This path helps define a feasible solution $x^*(\bar{\mathcal{S}}_t{}^-) + y^{r,SSK^*} - 1_{SSE,SSK} + 1_{SSE,l}$ to $\bar{\mathcal{S}}_t$. Since $x^*(\bar{\mathcal{S}}_t)$ is unique, we have $c_t \cdot (x^*(\bar{\mathcal{S}}_t{}^-) + y^{r,SSK^*} - 1_{SSE,SSK} + 1_{SSE,l}) < c_t \cdot x^*(\bar{\mathcal{S}}_t)$, provided the two solutions are distinct. Since $x_{rlt} = 1$ in $x^*(\bar{\mathcal{S}}_t)$ but not in the other solution, they are distinct. Therefore we have $C(\bar{\mathcal{S}}_t{}^-) + v_r(\bar{\mathcal{S}}_t{}^-) < C(\bar{\mathcal{S}}_t)$. By Corollary 4 we have $C(\bar{\mathcal{S}}_t) = C(\bar{\mathcal{S}}_t{}^-) + c_{rlt}$. Thus $v_r(\bar{\mathcal{S}}_t{}^-) < c_{rlt}$.

Since $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$ we now have $\hat{v}_{r,t+1} < c_{rlt}$.

Part b. Let $r' \in \mathcal{R}_t^{OPT}$ and suppose $r' \in \mathcal{R}_{t+1}^*$. We have that $x^*(\bar{\mathcal{S}}_t{}^- - \{r'\} \cup \{r\}) + 1_{r'l}$ is a feasible solution for $\bar{\mathcal{S}}_t$. Since $x^*(\bar{\mathcal{S}}_t)$ is unique, $c_t \cdot (x^*(\bar{\mathcal{S}}_t{}^- - \{r'\} \cup \{r\}) + 1_{r'lt}) < c_t \cdot x^*(\bar{\mathcal{S}}_t)$, provided the two solutions are distinct. Since $x_{rlt} = 1$ in $x^*(\bar{\mathcal{S}}_t)$ but not in the other solution, they are distinct. By Corollary 4 we have $C(\bar{\mathcal{S}}_t) = C(\bar{\mathcal{S}}_t{}^-) + c_{rlt}$. Thus $C^{r'-,r+}(\bar{\mathcal{S}}_t{}^-) + c_{r'lt} < C(\bar{\mathcal{S}}_t{}^-) + c_{rlt}$. By Shapley's Theorem we have

$$
\begin{aligned}
& C^{r+}(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) \leq C(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) + C^{r'-,r+}(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) \\
\Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t{}^-) \leq C(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) + C^{r'-,r+}(\bar{\mathcal{S}}_t{}^-) \\
\Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t{}^-) < C(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) + C(\bar{\mathcal{S}}_t{}^-) + c_{rlt} - c_{r'lt} \\
\Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t{}^-) - C(\bar{\mathcal{S}}_t{}^-) + c_{r'lt} < C(\bar{\mathcal{S}}_t{}^-) - C^{r'-}(\bar{\mathcal{S}}_t{}^-) + c_{rlt} \\
\Rightarrow \quad & v_r(\bar{\mathcal{S}}_t{}^-) + c_{r'lt} < v_{r'}(\bar{\mathcal{S}}_t{}^-) + c_{rlt} \\
\Rightarrow \quad & \hat{v}_{r,t+1} + c_{r'lt} < \hat{v}_{r',t+1} + c_{rlt}.
\end{aligned}
$$

(The last statement is due to the fact that $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t^-)$, by Case 2a, and $\hat{v}_{r',t+1} = v_{r'}(\bar{\mathcal{S}}_t^-)$, by Case 1bii.)

Since we have proven both cases, we have $x_t^{OPT} = x_t^*$. By induction, then, $x_t^{OPT} = x_t^* \ \forall t$, and therefore $x^{OPT} = x^*$.

We now prove the fixed-point property for the second problem class.

Theorem 4 *For a deterministic dynamic assignment problem in Problem Class 2, if the optimal solution is unique then it is a fixed point under our algorithm.*

Proof: The proof is similar to that for Theorem 3. Let x^* be the optimal solution, and let x^{OPT} be the solution produced by the gradients derived from the optimal solution. We wish to show that $x^{OPT} = x^*$. The proof will be by induction. As in Theorem 3, the base case $t = 0$ does not need to be dealt with explicitly.

Given time t there are three parts of the proof. We need to show that, under x^*:

1. If task l is not assigned during time t, then $\hat{v}_{r,t+1} > c_{rlt}$ for all r available and unassigned during t.
2. If task l is assigned during time t to r, then
   (a) $\hat{v}_{r,t+1} < c_{rlt}$, and
   (b) for any r' available and unassigned during t, $c_{r'lt} + \hat{v}_{r,t+1} < c_{rlt} + \hat{v}_{r',t+1}$.

Case 1b in Theorem 3 does not occur because the tasks are not persistent. Thus if task l is available during time t, $\mathcal{L}_t^{OPT} = \mathcal{L}_t^+ = \{l\}$, and there is no other task l' available to be assigned.

Given time t, suppose that $x_{t'}^{OPT} = x_{t'}^*$ for all $t' < t$. Then, since $\mathcal{R}_0^{OPT} = \mathcal{R}$, by the induction step we have $\mathcal{R}_t^{OPT} = \{r \in \mathcal{R} : x_{rlt'}^* = 0 \ \forall l, \forall t' < t\}$. Since tasks are not persistent, $\mathcal{L}_t^{OPT} = \mathcal{L}_t^+$. We wish to show that $x_t^{OPT} = x_t^*$. Let $l \in \mathcal{L}_t^{OPT}$. Either l is assigned under x^* or not.

Case 1. Suppose l is not assigned under x^*. Then $\mathcal{R}_t^* = \mathcal{R}_{t+1}^*$, as $\{l\} = \mathcal{L}_t^*$. Let $r \in \mathcal{R}_t^{OPT}$. Then $r \in \mathcal{R}_{t+1}^*$, as $\mathcal{R}_t^{OPT} = \mathcal{R}_t^*$ by induction hypothesis, and $\mathcal{R}_t^* = \mathcal{R}_{t+1}^*$, from above. Let $\bar{\mathcal{S}}_t^-, \bar{\mathcal{S}}_{t+1}$ be as in Corollary 5. Let $\bar{x_t}^{rl-}$ be the optimal assignment vector under $\bar{\mathcal{S}}_t^- - \{r, l\}$. Then $\bar{x_t}^{rl-}$ is

feasible for $\bar{\mathcal{S}}_t{}^-$, as $\bar{\mathcal{S}}_t{}^- - \{r,l\} \subset \bar{\mathcal{S}}_t{}^-$. Moreover, $\bar{x}_t{}^{rl-} + 1_{rl}$ (where 1_{rl} is a vector of zeros with a one in element (r,l)) is feasible for $\bar{\mathcal{S}}_t{}^-$. Thus by uniqueness we have

$$\begin{aligned} & c_t \cdot (x_t^{rl-} + 1_{rl}) < C(\bar{\mathcal{S}}_t{}^-) \\ \Rightarrow \quad & c_t \cdot x_t^{rl-} + c_{rlt} < C(\bar{\mathcal{S}}_t{}^-) \\ \Rightarrow \quad & C^{rl-}(\bar{\mathcal{S}}_t{}^-) + c_{rlt} < C(\bar{\mathcal{S}}_t{}^-). \end{aligned}$$

Consider the networks $\bar{\mathcal{S}}_t{}^- - \{r,l\}$ and $\bar{\mathcal{S}}_{t+1} - \{r\}$. Since tasks are not persistent, these networks have identical resource and task sets. Since $c_{r',l',t+1} \leq c_{r'l't}\ \forall r',\ \forall l'$, we have that, for any assignment x_{t+1} feasible for $\bar{\mathcal{S}}_{t+1} - \{r\}$, $c_{t+1} \cdot x_{t+1} \leq c_t \cdot x_{t+1}$. Thus we must have $C^{r-}(\bar{\mathcal{S}}_{t+1}) \leq C^{rl-}(\bar{\mathcal{S}}_t^-)$. Since l is not assigned under x^*, l is not assigned in the optimal solution for $\bar{\mathcal{S}}_t^-$ (see proof of Corollary 5). Thus, by Corollary 1, $C^{l-}(\bar{\mathcal{S}}_t^-) = C(\bar{\mathcal{S}}_t^-)$. Since $\bar{\mathcal{L}}_{t+1} = \bar{\mathcal{L}}_t^- - \{l\}$, by Corollary 5 and previous we have $C(\bar{\mathcal{S}}_{t+1}) = C(\bar{\mathcal{S}}_t{}^- - \{l\}) = C^{l-}(\bar{\mathcal{S}}_t^-) = C(\bar{\mathcal{S}}_t^-)$. Thus the above inequality gives us $C^{r-}(\bar{\mathcal{S}}_{t+1}) + c_{rlt} < C(\bar{\mathcal{S}}_{t+1})$. Since $\hat{v}_{r,t+1} = C(\bar{\mathcal{S}}_{t+1}) - C^{r-}(\bar{\mathcal{S}}_{t+1})$, we have $c_{rlt} < \hat{v}_{r,t+1}$.

Case 2. Suppose l is assigned under x^* to some resource r. Since tasks are transient, r and l must be assigned during time t.

Part a. First, we show that $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$. Let y^{r,SSK^*} be a maximal flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_{t+1}$. Since $l \notin \bar{\mathcal{S}}_t{}^-$ we have $\bar{\mathcal{L}}_t{}^- = \bar{\mathcal{L}}_{t+1}$. Thus, by the proof of Corollary 5, $\bar{\mathcal{S}}_t{}^-$ and $\bar{\mathcal{S}}_{t+1}$ have the same optimal assignments, and therefore y^{r,SSK^*} is a feasible flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t{}^-$. Either y^{r,SSK^*} contains a task unassigned in $\bar{\mathcal{S}}_{t+1}$ or not. If not, then by an argument similar to that in Case 1bii of Theorem 3 the contribution for each link in y^{r,SSK^*} in the network $\bar{\mathcal{S}}_{t+1}$ is the same as the contribution for the associated link in $\bar{\mathcal{S}}_t{}^-$. Since y^{r,SSK^*} is a feasible flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t{}^-$, we have $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t{}^-)$. If y^{r,SSK^*} contains a task l' unassigned in $\bar{\mathcal{S}}_{t+1}$, then in order to be feasible y^{r,SSK^*} must contain the mirror link (l', SSE) and a link (r', l') for some resource r'. Moreover, we can now see that y^{r,SSK^*} can contain at most one such unassigned task, as otherwise the supersource would occur twice in the path, creating a cycle and contradicting the tree of flow-augmenting paths described in Powell (1989). Thus y^{r,SSK^*} consists of a path from r to r' containing no unassigned tasks, the link (r', l'), the mirror link (l', SSE) and a path from SSE to SSK. We have already seen that the path from r to r' and the path from SSE to SSK contain no tasks unassigned in $\bar{\mathcal{S}}_{t+1}$; thus as in an earlier argument the contribution for each link in these paths is the same as that for the associated link in $\bar{\mathcal{S}}_t{}^-$. Since $c_{r'l',t+1} \leq c_{r'l't}$ we have that the length of y^{r,SSK^*} in $\bar{\mathcal{S}}_t{}^-$ is greater than or equal to

its length in $\bar{\mathcal{S}}_{t+1}$. And because y^{r,SSK^*} is feasible for $\bar{\mathcal{S}}_t^-$ this gives us $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t^-)$. In either case, then, $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t^-)$.

Now we show that $v_r(\bar{\mathcal{S}}_t^-) < c_{rlt}$. Let y^{r,SSK^*} be a maximal flow-augmenting path from r to SSK in $\bar{\mathcal{S}}_t^-$. This path helps define a feasible solution $x^*(\bar{\mathcal{S}}_t^-) + y^{r,SSK^*} - 1_{SSE,SSK} + 1_{SSE,l}$ to $\bar{\mathcal{S}}_t$. Since $x^*(\bar{\mathcal{S}}_t)$ is unique, we have $c_t \cdot (x^*(\bar{\mathcal{S}}_t^-) + y^{r,SSK^*} - 1_{SSE,SSK} + 1_{SSE,l}) < c_t \cdot x^*(\bar{\mathcal{S}}_t)$, provided the two solutions are distinct. Since $x_{rlt} = 1$ in $x^*(\bar{\mathcal{S}}_t)$ but not in the other solution, they are distinct. Therefore we have $C(\bar{\mathcal{S}}_t^-) + v_r(\bar{\mathcal{S}}_t^-) < C(\bar{\mathcal{S}}_t)$. By Corollary 4 we have $C(\bar{\mathcal{S}}_t) = C(\bar{\mathcal{S}}_t^-) + c_{rlt}$. Thus $v_r(\bar{\mathcal{S}}_t^-) < c_{rlt}$.

Since $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t^-)$ we now have $\hat{v}_{r,t+1} < c_{rlt}$.

Part b. Let $r' \in \mathcal{R}_t^{OPT}$ such that $r' \neq r$. Since r is assigned to l during t under x^*, $r' \in \mathcal{R}_{t+1}^*$. First, we show that $\hat{v}_{r',t+1} = v_{r'}(\bar{\mathcal{S}}_t^-)$. By the definition of the transition function, $\bar{\mathcal{S}}_{t+1} = \bar{\mathcal{S}}_t^-$ except that contributions between some resources and tasks could be higher in $\bar{\mathcal{S}}_t^-$. In particular, the contributions that change are exactly those between resources and tasks that are available both during times t and $t+1$ under x^*. But since tasks are not persistent, there are no resource/task pairs available during times t and $t+1$ under x^*. Thus the networks $\bar{\mathcal{S}}_{t+1}$ and $\bar{\mathcal{S}}_t^-$ are actually identical. And therefore $C(\bar{\mathcal{S}}_t^-) = C(\bar{\mathcal{S}}_{t+1})$, as well as $C^{r'-}(\bar{\mathcal{S}}_t^-) = C^{r'-}(\bar{\mathcal{S}}_{t+1})$. Thus,

$$\begin{aligned} v_{r'}(\bar{\mathcal{S}}_t^-) &= C(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) \\ &= C(\bar{\mathcal{S}}_{t+1}) - C^{r'-}(\bar{\mathcal{S}}_{t+1}) \\ &= \hat{v}_{r',t+1}. \end{aligned}$$

We have that $x^*(\bar{\mathcal{S}}_t^- - \{r'\} \cup \{r\}) + 1_{r'l}$ is a feasible solution for $\bar{\mathcal{S}}_t$. Since $x^*(\bar{\mathcal{S}}_t)$ is unique, $c_t \cdot (x^*(\bar{\mathcal{S}}_t^- - \{r'\} \cup \{r\}) + 1_{r'lt}) < c_t \cdot x^*(\bar{\mathcal{S}}_t)$, provided the two solutions are distinct. Since $x_{rlt} = 1$ in $x^*(\bar{\mathcal{S}}_t)$ but not in the other solution, they are distinct. By Corollary 4 we have $C(\bar{\mathcal{S}}_t) = C(\bar{\mathcal{S}}_t^-) + c_{rlt}$. Thus $C^{r'-,r+}(\bar{\mathcal{S}}_t^-) + c_{r'lt} < C(\bar{\mathcal{S}}_t^-) + c_{rlt}$. By Shapley's Theorem we have

$$\begin{aligned} & C^{r+}(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) \leq C(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) + C^{r'-,r+}(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) \\ \Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t^-) \leq C(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) + C^{r'-,r+}(\bar{\mathcal{S}}_t^-) \\ \Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t^-) < C(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) + C(\bar{\mathcal{S}}_t^-) + c_{rlt} - c_{r'lt} \\ \Rightarrow \quad & C^{r+}(\bar{\mathcal{S}}_t^-) - C(\bar{\mathcal{S}}_t^-) + c_{r'lt} < C(\bar{\mathcal{S}}_t^-) - C^{r'-}(\bar{\mathcal{S}}_t^-) + c_{rlt} \\ \Rightarrow \quad & v_r(\bar{\mathcal{S}}_t^-) + c_{r'lt} < v_{r'}(\bar{\mathcal{S}}_t^-) + c_{rlt} \\ \Rightarrow \quad & \hat{v}_{r,t+1} + c_{r'lt} < \hat{v}_{r',t+1} + c_{rlt}. \end{aligned}$$

(The last statement is due to the fact that $\hat{v}_{r,t+1} \leq v_r(\bar{\mathcal{S}}_t^-)$, by Case 2a, and $\hat{v}_{r',t+1} = v_{r'}(\bar{\mathcal{S}}_t^-)$, from above.)

Since we have proven both cases, we have $x_t^{OPT} = x_t^*$. By induction, then, $x_t^{OPT} = x_t^* \ \forall t$, and therefore $x^{OPT} = x^*$.

Acknowledgement

This research was supported in part by grant AFOSR-F49620-93-1-0098 from the Air Force Office of Scientific Research.

References

Powell, W. B. (1989), 'A review of sensitivity results for linear networks and a new approximation to reduce the effects of degeneracy', *Transportation Science* **23**(4), 231–243.

Shapley, L. S. (1962), 'Complements and substitutes in the optimal assignment problem', *Naval Research Logistics Quarterly* **9**, 45–48.

Spivey, M. Z. & Powell, W. B. (to appear), The dynamic assignment problem, *Transportation Science*.

Table 1: Results for Problem Class 1

Data Set Size	One Iteration		Convergence
	Optimal Start	Myopic Start	Myopic Start
5	100	100	100
10	100	99.0	99.0
15	100	100	100
20	100	98.3	100
25	100	100	100
30	100	98.7	99.7
35	100	99.4	100
40	100	96.5	99.7
45	100	99.8	99.8
50	100	98.8	99.6
55	100	99.4	99.6
60	100	95.2	95.2
65	100	99.8	99.8
70	100	98.0	98.4
75	100	96.9	96.9
80	100	99.4	99.7
85	100	98.3	99.7
90	100	95.5	99.9
95	100	99.0	99.3
100	100	98.5	98.5
MEAN	100	98.5	99.2
MEDIAN	100	98.9	99.7

Table 2: Results for Problem Class 2

Data Set Size	One Iteration		Convergence
	Optimal Start	Myopic Start	Myopic Start
5	100	100	100
10	100	99.0	99.0
15	100	100	100
20	100	98.3	100
25	100	100	100
30	100	95.1	97.7
35	100	99.4	100
40	100	86.8	99.7
45	100	99.8	99.8
50	100	95.5	99.6
55	100	99.4	99.6
60	100	95.2	95.2
65	100	96.8	99.8
70	100	95.5	99.4
75	100	96.9	96.9
80	100	99.4	99.7
85	100	98.3	99.7
90	100	95.3	99.7
95	100	97.5	99.3
100	100	98.5	98.5
MEAN	100	97.3	99.2
MEDIAN	100	97.9	99.7