Dynamic shortest path problem with travel-time-dependent stochastic disruptions : hybrid approximate dynamic programming algorithms with a clustering approach

Dynamic Shortest Path Problem with Travel-Time-Dependent Stochastic Disruptions: Hybrid Approximate Dynamic Programming Algorithms with a Clustering Approach

Derya Sever, Lei Zhao, Nico Dellaert, Tom Van Woensel, Ton de Kok

Beta Working Paper series 423

BETA publicatie	WP 423 (working paper)
ISBN	
ISSN	
NUR	804
Eindhoven	July 2013

Dynamic Shortest Path Problem with Travel-Time-Dependent Stochastic Disruptions: Hybrid Approximate Dynamic Programming Algorithms with a Clustering Approach

Derya Sever[a], Lei Zhao[b], Nico Dellaert[a], Tom Van Woensel[a], Ton De Kok[a]

[a]*Department of Operations Research and Accounting, Eindhoven University of Technology, Eindhoven, Netherlands*

[b]*Department of Industrial Engineering, Tsinghua University, Beijing, China*

Abstract

We consider a dynamic shortest path problem with stochastic disruptions in the network. We use both historical information and real-time information of the network for the dynamic routing decisions. We model the problem as a discrete time finite horizon Markov Decision Process (MDP). For networks with many levels of disruptions, the MDP faces the curses of dimensionality. We first apply Approximate Dynamic Programming (ADP) algorithm with a standard value function approximation. Then, we improve the ADP algorithm by exploiting the structure of the disruption transition functions. We develop a hybrid ADP with a clustering approach using both a deterministic lookahead policy and a value function approximation. We develop a test bed of networks to evaluate the quality of the solutions. The hybrid ADP algorithm with clustering approach significantly reduces the computational time, while still providing good quality solutions.

Keywords:
Dynamic shortest path problem, Approximate Dynamic Programming, Disruption handling, Clustering

1. Introduction

In traffic networks, disruptions due to accidents and traffic bottlenecks cause traffic congestions that lead to a drastic increase in travel times and decrease the probability of being on-time at the destination.

The travel time along a highway changes depending on the levels (types) of these disruptions. Furthermore, as the uncertainty in traffic networks increases due to recurrent and non-recurrent incidents, the planners need to take into account many levels of disruptions. Having real-time traffic information by using intelligent navigation systems and taking into account the stochastic nature of disruptions for the dynamic routing decisions can significantly reduce delays. As a trade-off, for networks with many levels of disruptions, computing dynamic routing decisions considering detailed information takes very long. Note that in real-life applications (as in navigation devices), low computation times enabling fast and high quality routing decisions are very important. Therefore, we need fast and efficient techniques to make the dynamic routing decisions considering stochastic disruptions.

In this paper, we consider dynamic shortest path problems with stochastic disruptions on a subset of arcs. We develop dynamic routing policies by using both historical information and real-time information of the network. We observe the disruption status of the network when we arrive to each node. The disruption status for the following stage is dependent on the travel time of the current arc. We denote this property as travel-time-dependency. We model the problem as a discrete time finite horizon Markov Decision Process (MDP). In our model, the dynamic routing policies are found based on the state of the system which can be retrieved with real time information.

MDP formulation provides a practical framework to find dynamic routing decisions for each decision epoch. However, for large scale networks with many levels of disruptions, obtaining the optimal solution faces the curses of dimensionality, i.e., states, outcomes, and decisions (Powell 2007 and Powell 2011). Therefore, in the dynamic shortest path problem literature with a MDP formulation, either approximation algorithms are developed or the structure of the optimal solution is investigated to deal with the curses of dimensionality .

The approximations or reduction techniques in the dynamic shortest path literature were shown to deal with binary levels of disruption, i.e. congested or not congested. The current empirical studies on the features of traffic congestion show that highways can have more than two levels of disruption which are specified according to the speed level and possible spillbacks (Helbing et al. 2009 and Rehborn et al. 2011). However, increase in disruption levels leads to an exponential state- and outcome-space growth causing the well-known curses of dimensionality. Therefore, we need efficient approximation techniques to deal with many levels of disruptions.

In this paper, we focus on the networks with many levels of disruptions which are similar to the real-life situations. We use an Approximate Dynamic Programming (ADP) algorithm which is a well-known approximation approach to effectively deal with the curses of dimensionality. In the literature, ADP is shown to be a powerful method to solve large-scale stochastic problems (e.g. Powell et al. 2002, Powell and Van Roy 2004, Simão et al. 2009).

The ADP algorithms used in this paper are based on value function approximations with a lookup table representation. First, we employ a standard value function approximation algorithm with various algorithmic design variations for updating state values with efficient exploration/exploitation strategies. Then, we improve the standard value function approximation algorithm by developing a hybrid ADP with a clustering approach. In this hybrid algorithm, we combine a deterministic lookahead policy with a value function approximation (Powell et al. 2012). We exploit the structure of disruption transition function to apply the deterministic lookahead policy. We form a cluster for each stage consisting of the nodes that are within the two-arcs-ahead neighborhood of the each intersection of roads. The decision is chosen from the cluster depending on a exploration/exploitation strategy. Then, we apply value function approximation until the destination.

Hybrid ADP algorithms with a lookahead policy and value function approximation is suggested by Powell et al. (2012). Similar approaches can be found in the literature on the hierarchal solution for large MDPs and using factored MDPs. In the factored representation of MDP, the states and/or decisions are clustered according to their specific properties. The value functions are then computed by using these compact representations (Boutilier et al. 2000, Givan et al. 2003, Kim and Dean 2002, Barry et al. (2011)).

In this paper, instead of clustering the states, we expand the action set from the neighboring nodes into the set of nodes in a cluster. By a learning process, the hybrid algorithm clusters the nodes such that the approximate value until the destination is lower depending on the current disruption status. The state variables that are considered in the value function approximation are then visited and updated more frequently while their state values become more accurate. This leads to higher quality approximate values for the state variables and reduces the computational cost significantly by eliminating the steps for updating and exploring the states that are already included in the lookahead policy. To our knowledge, the hybrid ADP algorithm with the clustering approach has not been investigated so far for the dynamic shortest path problems.

The main contributions of this paper are as follows:

1. We construct a hybrid ADP with a clustering approach considering a lookahead policy and a value function approximation to solve for the road networks with many levels of disruptions. The results show that the hybrid ADP algorithm with a clustering approach reduces the computational time significantly. Furthermore, the quality of the solution is higher compared to the standard ADP algorithm. We also provide a test bed consisting of random networks to compare the performance of the ADP algorithms with a stochastic lookahead policy shown to perform well in binary disruption levels.
2. We provide various algorithmic design variations for the ADP where multiple initial solutions and both single and double pass algorithms are used with efficient exploration/exploitation strategies. We provide insights about the performance of these variations based on different network structures.

2. Literature Review

The dynamic and stochastic shortest path problems are widely discussed in the literature, considering various solution approaches as adaptive routing, recourse policies (Fu 2001 and Polychronopoulos and Tsitsiklis 1996) and Markov decision process (MDP). The recourse approach updates the routes based on the realization of disruptions, while traveling through the network. The shortest paths are generated and updated during the trip as real-time information is retrieved from a navigation system. In this case, re-optimization is potentially executed after each retrieval of the online information.

Kim et al. (2005b) dealt with dynamic shortest path problems with anticipation using MDP for the optimal vehicle routing problem. Here, whenever there is new information about a disruption, the model takes into account the congestion dissemination and anticipates the route accordingly. For larger networks, as the formulation becomes intractable, Kim et al. (2005a) proposed state space reduction techniques where they identified the traffic data that has no added value in decision making process. Thomas and White (2007) also formulated the dynamic shortest path problem as an MDP. They provided conditions on which the optimal routing decision does not change even the network state changes. Bertsekas and Yu (2010) also formulated the stochastic shortest path problem as an MDP. For solving large scale problems, they developed a Q-learning algorithm with a policy iteration technique. They showed that their stochastic Q-learning algorithm is bounded and converges to the optimal at any initial solution.

Güner et al. (2012) considered non-stationary stochastic shortest path problem with both recurrent and non-recurrent congestions using real time traffic information. They formulated the problem as an MDP that generates dynamic routing policy based on the state of the system. To prevent the state explosion, they limit the formulation to two-links-ahead formulation where they only retrieve the state information for only two links ahead of the current location. Sever et al. (2013) also formulated the dynamic shortest path problems with travel-time-dependency using a MDP formulation. In this paper, the computational time is reduced by using different levels of the real-time, spatially dependent probabilistic disruption information and historical information for different parts of the network.

In this paper, we formulate the dynamic shortest path problem as an MDP. We apply ADP with value function approximation. We improve the standard ADP algorithm by developing a hybrid ADP algorithm with a clustering approach.

3. Markov Decision Process

We model the dynamic shortest path problem with stochastic disruptions as a discrete time finite horizon Markov Decision Process (MDP). Consider a traffic network represented by the graph G(N, A, A_v) where N represents the set of nodes (or intersections), A the set of directed arcs and A_v the set of vulnerable arcs, potentially in disruption ($A_v \subseteq A$). The number of vulnerable arcs is R: $R = |A_v|$. Each of these vulnerable arcs has a known unit-time transition probability matrix for all disruption levels. Each vulnerable arc, $r \in A_v$, can take any value from the disruption level vector, U^r, whose dimension depends on the specific vulnerable arc.

The travel time on an arc is assumed to be predictable from historical data and follows a discrete distribution, given the disruption level of the arc. We receive the real-time information about the disruptions for all vulnerable arcs when we arrive at a node. At each node, we make a decision on the next node to visit. Our objective is to travel from the origin node to the destination node with the minimum expected travel time. We derive the optimal routing decision using the MDP formulation with a finite number of stages, T. Stage t represents the number of nodes that have been visited so far from the origin node. The end of horizon, T, is reached by arriving at the destination node.

States

The state of the system at stage t, S_t, is represented by two components:

- The current node, $i_t \in N$.
- The disruption status vector, $\hat{D}_t$, gives the disruption statuses for all vulnerable arcs. Each vulnerable arc, r, can take any value from the disruption level vector U^r. For each vulnerable arc, there can be K_r different types of disruption levels: $\hat{D}_t(r) \in U^r$: $U^r = \{u^1, u^2, ...u^{K_r}\}, \forall r \in A_v$. Note that at each stage, we use a realization of the disruption vector, $\hat{D}_t$.

The state of the system at stage t is then: $S_t = (i_t, \hat{D}_t)$. We set the initial state as: $S_0 = (\text{origin node}, \hat{D}_0)$ and the final state as: $S_T = (\text{destination node}, \hat{D}_T)$ where $\hat{D}_0$ and $\hat{D}_T$ are the realizations of the disruptions for all vulnerable arcs at the initial and final stage, respectively.

Actions

Our action, x_t, is the next node to visit given the state S_t. We note that each action, x_t, is an element of the set of all possible actions $\mathcal{X}(S_t)$ which is the neighbor set of the current node:

$$x_t = i_{t+1} \tag{1}$$

Exogenous Information

The disruption statuses of the vulnerable arcs may change as we proceed to the next stage. The exogenous information consists of the realization of the disruption statuses of all the vulnerable arcs. Let $\hat{D}_{t+1}$ denote the disruption status realization that becomes known between stages t and $t+1$.

$$W_{t+1} = \hat{D}_{t+1} \tag{2}$$

Cost Function

The cost is calculated as the travel time from the current node, i_t to the next node, $x_t = i_{t+1}$, given the realized disruption status, $\hat{D}_t$:

$$C(S_t, x_t) \quad = \quad t_{i_t, x_t}(\hat{D}_t) \tag{3}$$

State Transition Function

At stage t, the system is in state S_t, we make a decision x_t and then observe the exogenous information W_{t+1}. The system transits to a new state S_{t+1} according to the transition function: S_{t+1}= $S^M(S_t, x_t, W_{t+1})$

The state transition involves the following transition functions:

$$i_{t+1} = x_t \tag{4}$$

$$D_{t+1} = \hat{D}_{t+1} \tag{5}$$

The disruption status vector transits from $\hat{D}_t$ to $\hat{D}_{t+1}$ according to a Markovian transition matrix. Note that D_{t+1} is the vector of random variables representing the disruption status of each vulnerable arc in the network for the next stage. We define the transition matrix for a vulnerable arc r from stage t to $t+1$ as $\Theta^r(t|S_t, x_t)$. This transition matrix is dependent on the state and the travel time of the current arc: the probability of being in the disruption status of the next stage $\hat{D}_{t+1}$, depends on the travel time between i_t and i_{t+1} given the disruption status realization, $\hat{D}_t$. Let $p^r_{u,u'}$ denote the unit-time transition probability between any two disruption levels for the vulnerable arc r, $p^r_{u,u'} = P\{\hat{D}_{t+1}(r) = u' | \hat{D}_t(r) = u\}$. $\Theta^r(t|S_t, x_t)$ is the transition vector for the vulnerable arc r from a disruption status realization, $\hat{D}_t(r) = u$, to a random disruption status at stage $t+1$, $D_{t+1}(r) = u'$.

$$\Theta^r(t|S_t, x_t) = \begin{bmatrix} p^r_{u,u^1} & p^r_{u,u^2} & \dots & p^r_{u,u^{K_r}} \end{bmatrix}^{t_{i_t,x_t}(\hat{D}_t)} \tag{6}$$

The probability of having the new disruption status $\hat{D}_{t+1}$ given $\hat{D}_t$ is then calculated as:

$$P(\hat{D}_{t+1}|\hat{D}_t) = \prod_{r=1}^{R} \Theta^r_{u,u'}(t|S_t, x_t) \tag{7}$$

Objective Function

The objective is to minimize the expected total travel time from the initial state until the final state:

$$\min_{\pi \in \Pi} \mathbb{E}(\sum_{t=1}^{T} C(S_t, \mathcal{X}^\pi(S_t))), \tag{8}$$

where π denotes a policy or decision rule and Π denotes the set of all policies.

4. Approximate Dynamic Programming Approach

The optimization problem in Equation (8) can be solved using the Bellman Equations:

$$V_t(S_t) \quad = \quad \min_{x_t \in \mathcal{X}_t} \quad C(S_t, x_t) + \sum_{S_{t+1}} P(S_{t+1}|S_t) V_{t+1}(S^M(S_t, x_t, W_{t+1})) \tag{9}$$

Here, $V_t(S_t)$ is the value of being in state S_t. Our solution becomes:

$$x_t^* = \arg\min_{x_t \in \mathcal{X}_t} \qquad C(S_t, x_t) + \mathbb{E}(V_{t+1}(S_{t+1})) \tag{10}$$

The MDP faces the curses of dimensionality with the increase in the number of disruption levels and the number of nodes. To solve large scale problems with many disruption levels, the dynamic programming approach for solving the Bellman's equations becomes computationally intractable. In the literature, many techniques such as state-reduction techniques (Kim et al. 2005a and Thomas and White 2007) and stochastic lookahead algorithms (Güner et al. 2012 and Sever et al. 2013) are used to solve for the MDP problems with reduced computational time. As an alternative the Approximate Dynamic Programming (ADP) approach is a powerful tool to overcome the curses of dimensionality, especially for complex and large scale problems (Powell 2007, Powell 2011).

In this paper, we use an ADP approach with value iteration (Powell 2007, Powell 2011). The essence of this approach is to replace the actual value function, $V_t(S_t)$, with an approximation which is denoted as $\bar{V}_t(S_t)$. Furthermore, instead of working backward through time as in the traditional DP approach, ADP works forward in time.

The optimization problem using the ADP approach is given in Equation (11):

$$\hat{v}_t^n = min \quad C(S_t^n, x_t^n) + \sum_{S_{t+1}} P(S_{t+1}|S_t)\bar{V}_{t+1}^{n-1}(S_{t+1}^M(S_t, x_t, W_{t+1}) \tag{11}$$

Post-decision State Variable

The value of being in state S_t^n, $\hat{v}_t^n$ contains the expectation over all possible states at the next stage. For large scale problems with many disruption levels, the state- and outcome-space explodes and computing the expectation can be computationally intractable. In other words, we can experience the curses of dimensionality due to state- and outcome-space and the expectation. Thus, in this paper, we adopt the post decision state variable as suggested by Powell (2007) and Powell (2011). We define the post-decision state as S_t^x which represents the state immediately after we make a decision. The post-decision is defined as follows:

$$S_t^x = S^{M,x}(S_t, x_t) = (x_t, \hat{D}_t) = (i_{t+1}, \hat{D}_t) \quad \text{(post-decision state)} \tag{12}$$

The post-decision state eliminates the expectation in Equation (11) by using the deterministic value of choosing an action, x_t, given the current state S_t.

Value Function Approximation with Lookup Table Representation

We use lookup table representation for the value function approximation. This means that for each discrete state S_t, we have an estimate $\bar{V}_t(S_t)$ which gives the approximate value of being in state S_t. We, then, update our estimate using a deterministic harmonic stepsize rule. As we use a post-decision state, we should update the value of being in the post-decision state of the previous stage by using $\hat{v}_t^n$.

$$\bar{V}_{t-1}^{n}(S_{t-1}^{x,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n \tag{13}$$

Exploration/Exploitation Strategy

In the ADP algorithm, we should decide on the trade-off between exploration and exploitation when making a decision given a certain state. If we only exploit, i.e., only choose the action with the minimum value, only the values of certain states with the minimum cost are updated, while the value of the rest of the states are not updated and remain at their initial values. This causes the approximate state values not improving as we do not explore other states. On the other hand, if we use exploration, we choose the action among an action vector and we reduce the probability of being stuck in suboptimal solutions.

In the early iterations, the value of being in a state is highly dependent on the sample path and the initial solution. Therefore, we need to use the exploration more in the early iterations to improve the quality of the state values. For this purpose, we use a mixed exploration and exploitation strategy (Powell 2007). To do this, we select a random probability of choosing the best action and choosing alternative actions. We ensure that the probability of choosing the best action increases as we visit the state more. Therefore, we choose to use exploration rate, ρ_t, for choosing the next best alternative as $\rho_t = b/n'(S_t)$. We should note that the exploration rate decreases with the number of visits to the particular state, $n'(S_t)$ and increases with the parameter "b".

Initialization Heuristics

According to Powell (2007) and Powell (2011), initial values play an important role for the accuracy of the approximate values and the computational performance of an ADP algorithm. For this purpose, in this paper, we investigate the effect of two initialization heuristics: a deterministic approach and a stochastic two-arcs-lookahead policy with memoryless probability transitions (denoted as $DP(2, M)$).

In the deterministic approach, we determine initial state values by solving a deterministic shortest path problem, assuming that the probability of having a disruption is zero. In determining the route, we only consider the non-disruption state for all vulnerable arcs. In the transition probabilities for all vulnerable arcs, the no-disruption state becomes an absorbing state in Equation (6). Then, we solve Equation (9). The output from this initialization heuristics is the same initial value for all the state variables.

As the dynamic shortest path problem is travel-time-dependent, including disruption status dependent initial values in the ADP algorithm may improve the quality of the solutions. For large size instances, solving for the stochastic shortest path problem with the memoryless probability transitions for all vulnerable links in the network can be computationally challenging. Therefore, we use a lookahead policy. The stochastic two-arcs-lookahead policy with memoryless probability transitions, $DP(2, M)$ considers the memoryless probability transitions for the arcs that are only two-arcs-ahead from the current node. We calculate the expected travel time until the destination for the arcs that are beyond the two-arcs-ahead neighborhood of the current node.

In this policy, we have online information for only two-arcs-ahead from the current node. Therefore, the state space for the current node i_t is modified as $S_t = (i_t, \hat{D}_t^{i_t})$ where $\hat{D}_t^{i_t} = \{u_t^{r_{i_t 1}}, u_t^{r_{i_t 2}}, .., u_t^{r_{i_t K^r_{i_t}}}\}$ where with r_{i_t} which is the vector of the vulnerable arcs that are two-arcs-ahead neighborhood of the node i_t and $R_{i_t} = |r_{i_t}|$, $R_{i_t} \leq R$. For the rest of the arcs, we calculate expected travel times. The transition probability vector, in Equation (6) is no longer travel-time-dependent and we only consider the limited part of the transition matrix where only the vulnerable arcs that are maximum two-arcs-ahead from the current node is included. The output from this initialization heuristics is an initial value for each state variable.

Updating the value function

In designing an ADP algorithm, an important decision to make is the way we update the approximate value. Two different approaches are developed for updating the value function: single-pass and double-pass. In this paper, we investigate the effect of both methods.

In this paper, we first present the standard ADP algorithm using the algorithmic variations described above. Then, we develop a hybrid ADP with a clustering approach by combining a deterministic lookahead policy with the value function approximation (Powell et al. (2012)). In the following sections, we first describe the standard ADP algorithm and then the hybrid ADP algorithm with a clustering approach.

4.1. The Standard Approximate Dynamic Programming Algorithm

In the standard ADP, we adopt the value function approximation algorithm with a post-decision state variable. We use a mixed exploration and exploitation strategy to update the value function approximations. Two different approaches are developed for updating the value function: single-pass and double-pass.

ADP Algorithm: Single-Pass Version (ADP_S)

In the ADP algorithm with single-pass, updating the value function takes place as the algorithm progresses forward in time. According to the Algorithm 4.1, in Step 2a, at time t, we obtain an updated estimate of the state value of being in state S_t. In step 2b, we then update the estimate of $\bar{V}^n_{t-1}$ by using the state value from the previous iteration and the current value estimate $\hat{v}^n_t$. Note that we collect information as we proceed in time due to the fact that the disruption status of the future stages is dependent on the current state.

ADP Algorithm: Double-Pass Version (ADP_D)

As an alternative to the single-pass value iteration algorithm, we also use a double-pass algorithm (Algorithm 4.2). In this algorithm, we step forward in stages by creating a trajectory of states, actions and

Algorithm 4.1 ADP Algorithm with Single-Pass

```
Step 0: Initialization
    Step 0a: Initialize $\bar{V}_t^0$, $\forall t$ by using the value from the initialization heuristic.
    Step 0b: Set n=1.
Step 1: Choose a sample disruption vector with Monte Carlo simulation for $n$ and for $t = 0$, $\omega_0^n$ and initialize $S_0^n = (start\_node, \hat{D}_0^n)$.
Step 2: Do for all $t = 0, ..., T-1$ (where $T$ is reached by arriving to the destination node)
    Step 2a: Choose a random probability $p$ and find an exploration rate $\rho_t$ as $\rho_t = 0.2/n'$
        $\hat{v}_t^n = \min_{x_t \in I, x_t \in \mathcal{X}_t} C(S_t^n, x_t^n) + \bar{V}_t^{n-1}(x_t, \hat{D}_t^n)$   (14)
        The node $x_t$ that gives the optimal value is denoted as $x_t^{*n}$
        $\hat{v}_t^n = \begin{cases} \hat{v}_t^{*n} & \text{if} \quad p \geq \rho_t \\ \text{solve Equation 14 for } x_t \neq x_t^{*n} & o.w \end{cases}$
        The node that is chosen is denoted as $x_t^n$ and becomes the next node to visit.
    Step 2b: If $t > 0$, update $\bar{V}_{t-1}^n(S_{t-1}^{x,n})$ using:
        $\bar{V}_{t-1}^n(S_{t-1}^{x,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n$   (15)
        Where we use harmonic stepsize: $\alpha_{n-1} = \frac{a}{a+n'(S_t)-1}$ and $n'(S_t)$ is the total number of visits to state $S_t$.
    Step 2c: Find the post-decision state: $S_t^{x,n} = (x_t, \hat{D}_t^n)$
    Step 2d: Find the next pre-decision state: $S_{t+1}^n = (x_t, W_{t+1}^n)$
Step 3: Increment $n$. If $n \leq \mathbb{N}$ go to Step 1.
Step 4: Return the value functions $(\bar{V}_t^N)_{t=1}^T$
```

outcomes (Step 2). Then, we update the value function by stepping backwards through the stages (Step 3). The update of the value function is conditional on whether we exploit or explore. If we exploit at stage t (choosing the action with the minimum value), then we update the value function of the state at stage t. If we choose exploration, then we update the value function if the chosen action improves the value function compared to the exploitation (Step 3b). Otherwise, we do not update.

Algorithm 4.2 ADP Algorithm with Double-Pass

```
Step 0: Initialization
Step 1: Choose a sample disruption vector with Monte Carlo simulation for $n$ and for $t = 0$, $\omega_0^n$ and initialize $S_0^n = (start\_node, W_0^n)$.
Step 2: Do for all $t = 0, 1, ..., T-1$ (where $T$ is reached by arriving to the destination node)
  Step 2a: Solve:
      $x_t^{*n} = \underset{x_t \in I, x_t \in \mathcal{X}_t}{\operatorname{argmin}} \quad C(S_t, x_t) + \bar{V}_t^{n-1}(x_t, \hat{D}_t^n)$    (16)
      $x_t^n = \begin{cases} x_t^{*n} & \text{if} \quad p \geq \rho_t \\ \text{solve Equation 16 for } x_t \neq x_t^{*n} & o.w \end{cases}$
  The node $x_t$ that gives the optimal value is denoted as $x_t^{*n}$ and $x_t^{\Delta n}$ if it is from the exploration.
  Step 2b: If we choose exploration with $x_t^{\Delta n}$ find :
      $\bar{V}_{t-1}^{*n}(S_{t-1}^{x,n}) = (1 - \alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^{*n}$    (17)
      $\hat{v}_t^{*n} = C(i, x_t^{*n}, \hat{D}_t^n) + \bar{V}_t^{n-1}(x_t^{*n}, \hat{D}_t^n)$    (18)
  Step 2c: Find the post-decision state
  Step 2d: Find the next pre-decision state
Step 3: Do for all $t = T-1, ..., 1$
  Step 3a: Compute $\hat{v}_t^n$ using the decision $x_t^n$ from the forward pass:
      $\hat{v}_t^n = C(S_t^n, x_t^n) + \hat{v}_{t+1}^n$    (19)
  Step 3b: If $t > 1$, update $\bar{V}_{t-1}^n(S_{t-1}^{x,n})$ using:
  If we have explored at time $t$ with action $x_t^{\Delta n}$, let:
      $\bar{V}_{t-1}^{\Delta n}(S_{t-1}^{x,n}) = (1 - \alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n$    (20)
  Then update the value function by:
      $\bar{V}_{t-1}^n(S_{t-1}^{x,n}) = \begin{cases} (1 - \alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n & \text{if} \quad x_t^{*n} \\ (1 - \alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha_{n-1}\hat{v}_t^n & \text{if} \quad x_t^{\Delta n} \& \bar{V}_{t-1}^{\Delta n}(S_{t-1}^{x,n}) < \bar{V^*}_{t-1}^{n}(S_{t-1}^{x,n}) \end{cases}$
  Where we use harmonic stepsize: $\alpha_{n-1} = \frac{a}{a + n'(S_t) - 1}$ and $n'(S_t)$ is the number of visits to the current state.
Step 4: Increment $n$. If $n \leq \mathbb{N}$ go to Step 1.
Step 5: Return the value functions $(\bar{V}_t^N)_{t=1}^T$
```

4.2. Hybrid ADP with Clustering Approach : a Deterministic lookahead policy with Value Function Approximations

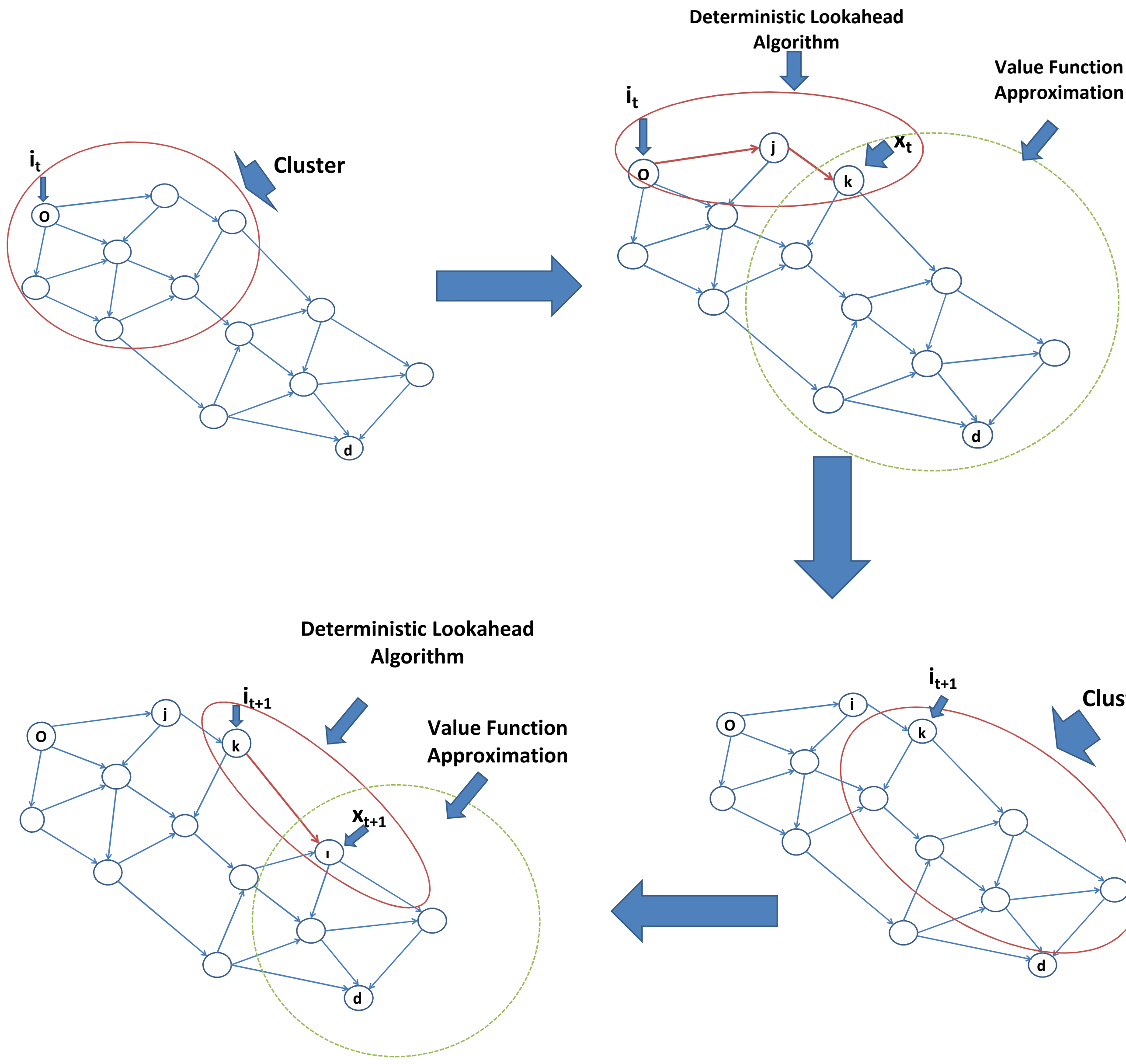
The state variable in our MDP formulation is a representation of the nodes and the disruption statuses for the vulnerable arcs. When the network size and the number of disruption levels increases, the number of state and value function approximations in the lookup table also increases. To improve the solution quality and to reduce the computational time for large size instances, we need to improve the standard ADP algorithm. To do this, we exploit the structure of the disruption transition function in Equation (6) which is travel-time-dependent. The structure of the disruption transition function shows that the shorter the travel time between two stages, the disruption status of the vulnerable arcs at the next stage is similar to the disruption status at the current stage. In other words, in a close neighborhood of the current node, i_t, the disruption status of the observed vulnerable arcs from the stage t hardly changes at the stage $t + 1$.

This structure is a good motivation to cluster the homogeneous neighborhoods of nodes that are close to each other. Using this clustering idea, we develop a hybrid ADP algorithm. In this hybrid algorithm, we first apply a deterministic lookahead policy for the cluster and then we use the value function approximation until the destination.

By a learning process, the hybrid algorithm clusters the nodes such that the approximate value until the destination is lower depending on the current disruption status. The state variables that are considered in the value function approximation are then visited and updated more frequently while their state values become more accurate. This leads to higher quality approximate values for the state variables and reduces the computational cost by eliminating the steps for updating the states that are already included in the lookahead policy. In this way, we achieve an efficient exploration/exploitation process with more accurate approximations.

Figure 1 illustrates how we do the clustering and perform the hybrid ADP algorithm. At stage t, we have the state variable $S_t = (i_t, \hat{D}_t)$. We form the cluster of the current node, i_t, by simply considering the nodes that are two-arcs-ahead from i_t. We denote the cluster of i_t as: cl_{i_t}. Because we assume that the disruption status, $\hat{D}_t$ does not change within two-arcs-ahead, we apply a deterministic lookahead policy

Figure 1: The Hybrid lookahead policy Value Function Approximation Process


Cluster
i_t
O
d
Deterministic Lookahead Algorithm
Value Function Approximation
i_t
O
j
k
x_t
d
Cluster
i_{t+1}
O
i
k
d
Deterministic Lookahead Algorithm
Value Function Approximation
i_{t+1}
O
j
k
ι
x_{t+1}
d

within the cluster. The lookahead policy consists of solving the Dijkstra's deterministic shortest path algorithm (Dijkstra 1959) for the cluster. In this way, we determine the shortest path from i_t until the next node i_{t+1} which is in the current cluster cl_{i_t}. The cost from the current node to the selected next node given the disruption status is determined by the Dijkstra's shortest path algoritm and it is denoted as $\tilde{C}_t(i_t, x_t, \hat{D}_t)$.

After finding the next node to visit with the exploration/exploitation strategy, we then approximate the current state value, S_t using a harmonic stepsize rule. Then, we continue to find the next cluster consisting of the two-arcs-ahead nodes of the next node and approximate its value. We always go further towards the destination node. We apply the same process until the destination node (Figure 1).

The hybrid ADP algorithm with the clustering approach has three types of algorithmic variations: by formulation of the value function, by deciding on which nodes to update when we determine a shortest path and whether we use single-pass or double-pass.

Hybrid ADP with Clustering Approach : a Deterministic Lookahead Policy with Value Function Approximations-Type 1 (C_1)

In the hybrid ADP with the clustering approach type 1, the clusters are formed by considering the nodes that are two-arcs-ahead from the current node. We apply the deterministic lookahead policy to obtain the cost between i_t and x_t, $\tilde{C}_t(i_t, x_t, \hat{D}_t^n)$. We, then add the minimum value of the travel time from node x_t to the next node j given the current disruption status (where j can be within the cluster or outside of the cluster) and the approximate value function from the node j to the destination node. Here, we should note that we always ensure that we take the deterministic travel time within the cluster. Note that by this formulation actually, we consider the clustering of the nodes that are three-arcs-ahead from the current node. In this way, we also investigate the effect of the size of the clustering horizon.

One of the variations of the hybrid algorithm is to decide whether to update the state values of the nodes within the shortest path or not. For each shortest path, we obtain a path consisting of nodes to travel until x_t. So, we also investigate whether updating the state values of the path obtained from the lookahead policy improves our solution or not. For instance, our current node is i_t and we choose to go to node x_t by using the shortest path $i_t - z - x_t$. If we only update the state value of node i_t, we call this as "No-Update", NU. If we also update the state value of node z, then we call this as "Update", U (Note that as in standard double pass algorithm, we update the state value of path nodes if they improve our approximate value function).

For the single-pass algorithm with Clustering 1 approach with or without updating of the deterministic shortest path ($C_1 - ADP_{S,U}$, $C_1 - ADP_{S,NU}$), we replace Equation (14) with the equation below:

$$\hat{v}_t^{*n} = \min_{x_t \in cl_{i_t}} \quad \tilde{C}_t(i_t, x_t, \hat{D}_t^n) + \min_{j \in Neighbor(x_t)} \{C(x_t, \hat{D}_t^n, j) + \bar{V}_t^{n-1}(j, \hat{D}_t^n)\} \tag{21}$$

For the double-pass algorithm with Clustering 1 approach with or without updating of the deterministic shortest path ($C_1 - ADP_{D,U}$, $C_1 - ADP_{D,NU}$), we replace Equation (16) with the equation below:

$$x_t^{*n} = \operatorname*{argmin}_{x_t \in cl_{i_t}} \quad \tilde{C}_t(i_t, x_t, \hat{D}_t^n) + \min_{j \in Neighbor(x_t)} \{C(x_t, \hat{D}_t^n, j) + \bar{V}_t^{n-1}(j, \hat{D}_t^n)\} \tag{22}$$

Hybrid ADP with Clustering Approach : a Deterministic Lookahead Policy with Value Function Approximations-Type 2 (C_2)

As in C_1, we apply the deterministic lookahead policy to obtain the cost between i_t and x_t, $\tilde{C}_t(i_t, x_t, \hat{D}_t^n)$ for the cluster of nodes that are two-arcs-ahead from the current arc. We, then add the value function approximation from x_t to the destination node. Compared to clustering type 1, this approach considers clustering only the nodes that are two-arcs-ahead from i_t.

For the single-pass algorithm with Clustering 2 approach with or without updating of the deterministic shortest path ($C_2 - ADP_{S,U}$, $C_2 - ADP_{S,NU}$), we replace Equation (14) with the equation below:

$$\hat{v}_t^{*n} = \min_{x_t \in cl_{i_t}} \quad \tilde{C}_t(i_t, x_t, \hat{D}_t^n) + \bar{V}_t^{n-1}(x_t, \hat{D}_t^n) \tag{23}$$

For the double-pass algorithm with Clustering 2 approach with or without updating of the deterministic shortest path ($C_2 - ADP_{D,U}$, $C_2 - ADP_{D,NU}$), we replace Equation 16 with the equation below:

$$x_t^{*n} = \operatorname*{argmin}_{x_t \in cl_{i_t}} \quad \tilde{C}_t(i_t, x_t, \hat{D}_t^n) + \bar{V}_t^{n-1}(x_t, \hat{D}_t^n) \tag{24}$$

5. Dynamic Programming with Stochastic Two-arcs-lookahead policy ($DP(2,H)$)

As the number of disruption levels and network size increases, the MDP becomes computationally intractable. At large instances, we cannot compare the performance of the ADP algorithms with the optimal algorithm via a MDP. Therefore, we need another benchmark heuristic for the large size instances. We provide a dynamic programming approach with two-arcs-lookahead policy, ($DP(2,H)$) that was developed in Sever et al. (2013). This algorithm is reasonable to use as a benchmark, because it is shown that it performs only slightly worse than the optimal algorithm. This algorithm considers limited online information for each stage. The intuition behind this is that due to the structure of the disruptions (where the probability of experiencing the disruption decreases within time), by the time we arrive to the further arcs, we will experience the steady-state probabilities. Therefore, we eliminate the computational burden to calculate the transition probabilities for all vulnerable arcs.

In this policy, we have online information for two-arcs-ahead from the decision node. Therefore, the state space of the hybrid policy for any current node i_t is modified as $S_t = (i_t, \hat{D}_t^i)$ where $\hat{D}_t^i = \{u_t^{r_{i_t 1}}, u_t^{r_{i_t 2}}, .., u_t^{r_{i_t R_{i_t}}}\}$ with r_{i_t} which is the vector of the vulnerable arcs that are two-arcs-ahead neighborhood of the node i_t and $R_{i_t} = |r_{i_t}|$, $R_{i_t} \leq R$. For the rest of the arcs, we calculate the memoryless distributions.

Given the state is S_t, a transition is made to the next decision state, $S_{t+1} = (x_t, \hat{D}_{t+1}^{x_t})$. For hybrid policies, we use the travel-time-dependent state transition matrix for the vulnerable arcs for which we have online information. The probability distribution of being in state S_{t+1} from the state S_t given travel-time-dependent transition matrix is the same as the one give in Equation (6) except we only consider the limited part of the transition matrix where only the vulnerable arcs that are maximum two-arcs-ahead from the current node is included.

6. Computational Experiments and Analysis

In this paper, for our experimental design, we evaluate and compare the algorithms that are summarized with following variations:

	Single/Double-Pass	Update	Clustering
ADP_S	Single	NA	No
ADP_D	Double	NA	No
$C_1 - ADP_{(S,NU)}$	Single	No-Path-Update	Type 1
$C_1 - ADP_{(D,NU)}$	Double	No-Path-Update	Type 1
$C_1 - ADP_{(S,U)}$	Single	Path-Update	Type 1
$C_1 - ADP_{(D,U)}$	Double	Path-Update	Type 1
$C_2 - ADP_{(S,NU)}$	Single	No-Path-Update	Type 2
$C_2 - ADP_{(D,NU)}$	Double	No-Path-Update	Type 2
$C_2 - ADP_{(S,U)}$	Single	Path-Update	Type 2
$C_2 - ADP_{(D,U)}$	Double	Path-Update	Type 2

In the experimental study, we fix the parameters for algorithmic variations of the ADP algorithms. We get initial state values from the solution of the deterministic approach and from the two-arcs-lookahead dynamic shortest path problem with memoryless transitions. After performing several preliminarily tests, we set the constant in the harmonic stepsize rule to $a = 5$ for a reasonable convergence and higher quality of solutions. We also set the exploration rate for choosing random alternative decisions as $0.2/n'(S_t)$ where $n'(S_t)$ is the number of visits to the state. Also, we fix the iteration counter to $N = 100000$ for the ADP algorithms to ensure the convergence.

In what follows, we describe our test instances. Then, we present the results of the ADP algorithms with different algorithmic design variations. Finally, we compare the algorithms's performances with respect to solution quality and computational time.

6.1. Design of Test Instances

We generate different network types using a number of network properties to characterize[1]. These network properties are as follows:

[1]All instances and their detailed results are available from the authors upon request

Network Size The network consists of small, medium and large size networks with 16, 36 and 64 nodes, respectively. The network is designed such that the origin-destination pairs are situated from the top-left to the bottom-right corners. With this structure, we prevent evaluating unnecessary nodes far from the shortest path. Clearly, this does not limit the applicability of our results, but merely reduces the number unnecessary calculations for further nodes.

Number of Vulnerable arcs Vulnerable arcs are randomly assigned to the shortest paths that are found with a dynamic process every time a new vulnerable arc is added to the network. We have instances with low and high numbers of vulnerable arcs. Specifically, 50% (low number) or 80% (high number) of the arcs in the shortest path are labeled to be vulnerable.

Number of Disruption Levels For each vulnerable arc the possible disruption levels are kept the same and there can be K different types of disruption levels. In this experimental study, we have 2, 3 and 5 levels of disruptions.

Travel Times Each arc has a randomly selected travel time taken from a discrete uniform distribution $U[1, 10]$. If there is a disruption, the travel time length is multiplied with 3 at most. If there are more than 2 disruption levels, we keep the same expected travel time for various number of disruption levels.

Probability of having a disruption on the arc Define a lower probability of having disruptions to be between $[0.1 - 0.5)$ and a higher probability between $[0.5 - 0.9)$.

We define "network type" as the network that has specific network dimensions. For instance, small size network with low number of vulnerable arcs with low disruption probability is a unique network type. In this paper, we generate 36 different network types with the relevant properties as seen in Table 1. For each of the network type, we randomly generate 50 instances with random travel times and random locations for vulnerable arcs. So, in total we generate 1800 test instances.

Table 1: Network Dimensions

Number of Nodes	Number of vulnerable arcs		Probability of having a disruption		Number of disruption levels
	Low	High	Low	High	
16	3	5	[0.1- 0.5)	[0.5- 0.9)	2, 3, 5
36	5	7	[0.1- 0.5)	[0.5- 0.9)	2, 3, 5
64	7	9	[0.1- 0.5)	[0.5- 0.9)	2, 3, 5

6.2. Evaluation of the Algorithms

We analyze the performance of the standard ADP algorithms, the Hybrid ADP algorithms with type 1 and 2, the dynamic programming algorithm with stochastic two-arcs-lookahead policy ($DP(2, H)$) and the optimal algorithm via a MDP. For the evaluation of the algorithms, we use two procedures: an exact evaluation and an evaluation via a simulation. We perform an exact evaluation for the networks where the optimal algorithm is computationally tractable (N=16). For instance, for the networks with lower levels of disruption and lower number of vulnerable arcs, the optimal algorithm is tractable. In the exact evaluation, for each algorithm, we obtain routing policies considering each possible state. Then, we compute the exact value function using the Equation (9) with these pre-determined policies for each instance. This value gives the expected cost of the algorithm considering all possible states.

For the networks where the optimal algorithm is no longer tractable, we simulate each algorithm with a sample size of 5000 for each network instance (N > 16). We ensure that each algorithm uses the same sample of transition probabilities for each instance. For each network instance, we find the average travel time and computational time by calculating the average values of the simulations.

For each type of the network, we run 50 different instances and then provide the average travel time and the percentage gap with respect to the benchmark heuristic, i.e. $DP(2, H)$. Note that we choose the $DP(2, H)$ algorithm as a benchmark because the optimal policy can only be obtained for the network instances with lower levels of disruptions and low vulnerability.

6.3. Experimental Results and Discussion

In this section, we report on the computational experiments that implement the ADP algorithms, the benchmark heuristic and the optimal algorithm. We report the experimental results by aggregating them according to these network properties: the disruption rate, the network size, the vulnerability of the network and the number of disruption levels. We first analyze the effect of using different algorithmic properties for the ADP algorithms. Then, we compare the performance of the hybrid ADP algorithms with the optimal algorithm, the standard ADP algorithm and the DP(2,H). We will ground our analysis by comparing their expected travel times, the percentage error with respect to the benchmark heuristic and the computational time.

The algorithms presented in this paper are programmed in Java. All experiments are conducted on a personal computer with an IntelCore Duo 2.8 Ghz Processor and 3.46 GB RAM.

Effect of Algorithmic Properties of The ADP Algorithms

We compare the performance of the ADP algorithms that use different algorithmic properties by providing the expected travel times for only the small and medium size networks with 2 and 3 disruption levels.

Effect of Initialization Heuristics

In the ADP algorithms, we use two different initialization heuristics for obtaining the initial value function estimates: a deterministic approach and a stochastic two-arcs-lookahead policy with memoryless probability transitions ($DP(2, M)$). In the ADP, the value function estimate, $\bar{V}_t^{n-1}$ used in iteration n affects the choice of the action taken and the next state visited. Therefore, it is important to choose an efficient initialization heuristic such that we have accurate value function estimates.

Figure 2 shows the expected travel times from the ADP algorithms using these initial values for small and medium size networks with 2 and 3 disruption levels. When there are 2 disruption levels, the ADP algorithms using deterministic initial values perform slightly worse than or similar to the ADP algorithms using $DP(2, M)$ as the initialization heuristic. However, as the number of disruption levels and the network size increases, the ADP algorithms with the deterministic initial values perform much worse. This indicates that when we have more states, starting with the state-dependent initial values (as in $DP(2, M)$) provides more efficient exploration and exploitation such that we obtain higher quality value function estimates. In the deterministic initial solutions, we start with lower and same approximate values for all states. Then, the higher cost paths are chosen and updated initially such that we may stuck into suboptimal solutions.

For the rest of the numerical analysis, we adopt $DP(2, M)$ as the initialization heuristic for the ADP algorithms because it gives higher solution quality.

Single-Pass versus Double-Pass

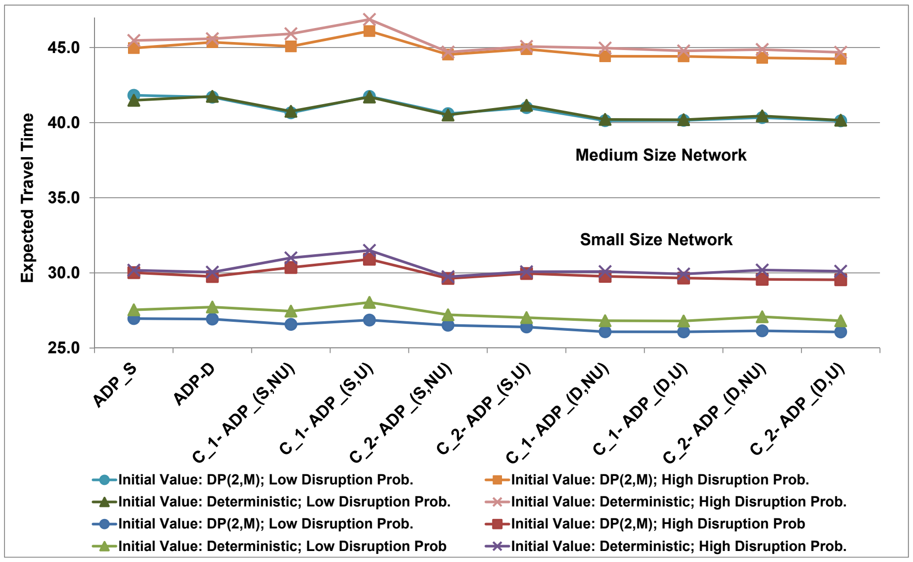
To investigate the effect of the way we update the value functions, we both use single-pass and double-pass procedures in the ADP algorithms. Figure 3 and Figure 4 illustrate the expected travel times for both the Standard ADP and the Hybrid ADP algorithms for different network sizes and different number of vulnerable arcs respectively.

Figure 3 shows that double-pass approach for the standard ADP algorithms does not perform well on the large size networks. While both of the algorithms perform similarly in the small and the medium size networks. Moreover, in the double-pass algorithm (Algorithm 4.2), the update is done via a backward pass, the frequent change in actions and paths may prevent the algorithm to give better estimates.

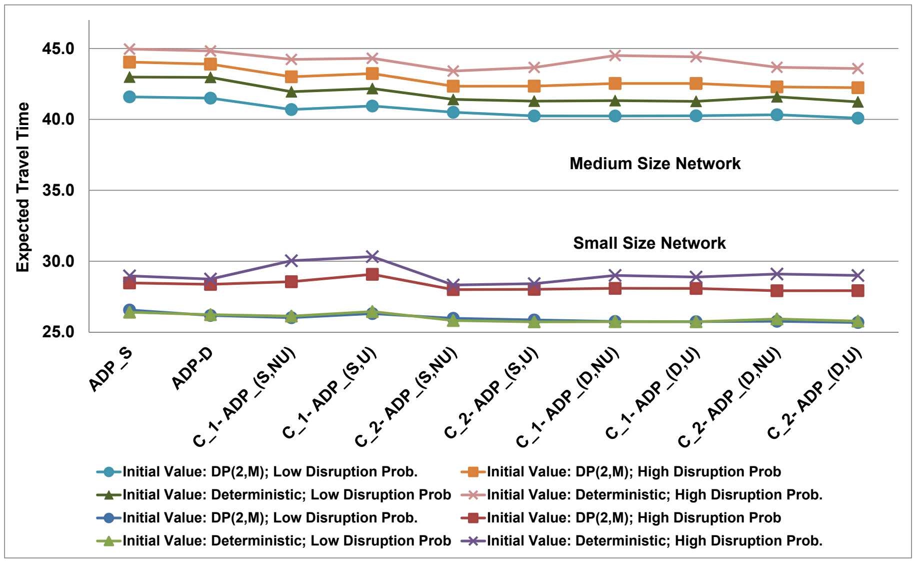
On the other hand, for the hybrid ADP algorithms with the clustering approach, double-pass algorithms generally perform better than the single-pass algorithms.

The effect of using single-pass and double-pass approaches in our ADP algorithms is even more visible when we aggregate the networks according to their number of vulnerable arcs. Figure 4 shows that ADP_D performs worse than ADP_S in the networks with two and three disruption levels. For the hybrid ADP algorithms with the clustering approach, double-pass algorithms again perform better than the single-pass algorithms. In the figure, it is shown that $C_1 - ADP_{(S,.)}$ and $C_2 - ADP_{(S,.)}$ performs worse than $C_1 - ADP_{(D,.)}$ and $C_2 - ADP_{(D,.)}$ algorithms.

Figure 2: The expected travel time from the ADP algorithms with Deterministic and DP(2,M) initial solutions for different network sizes with 2 and 3 disruption levels

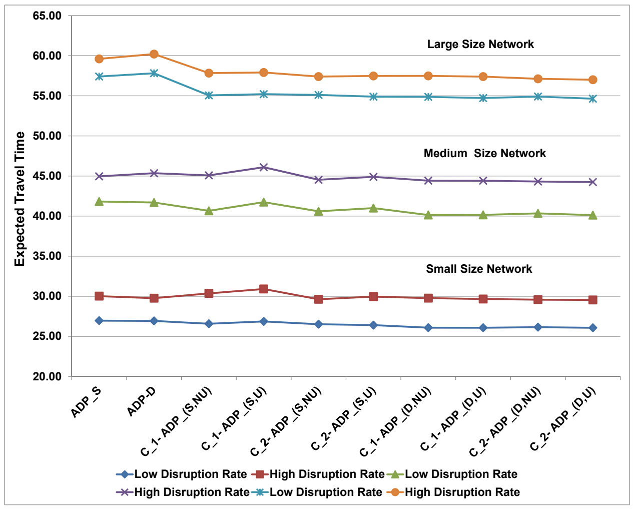


2 Disruption Levels

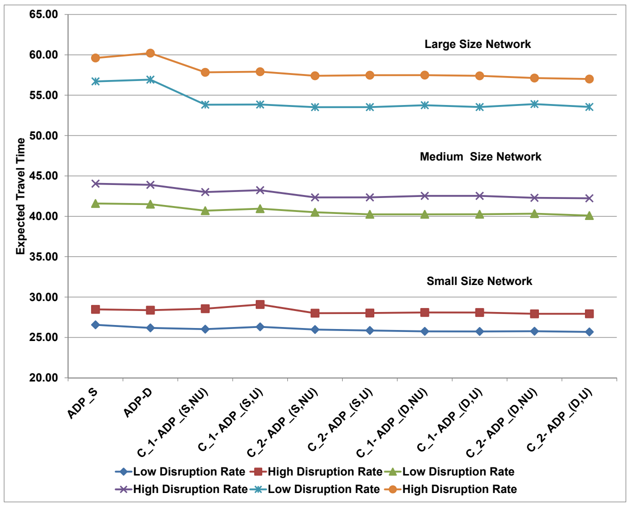


3 Disruption Levels

Figure 3: The expected travel time from ADP algorithms for different network sizes with 2 and 3 disruption levels



2 Disruption Levels



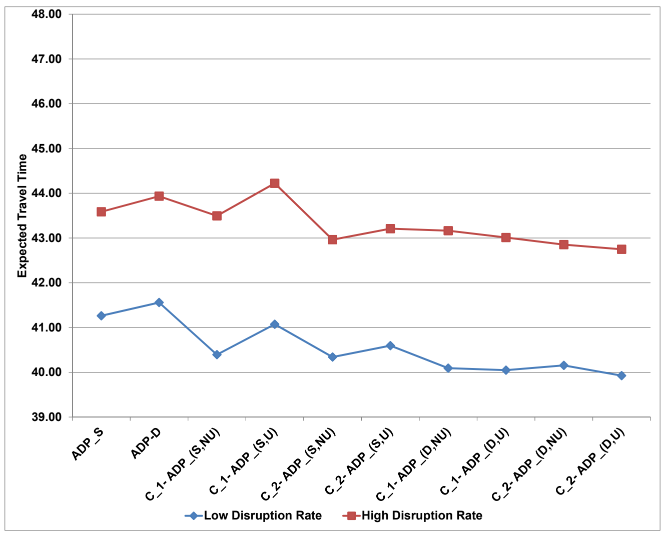
3 Disruption Levels

The performance difference between $C_1{-}ADP_{(S,.)}$ and $C_1{-}ADP_{(D,.)}$ is significantly high. In $C_1{-}ADP_{(S,.)}$, we perform a deterministic lookahead policy for the three-arcs-lookahead neighborhood. So, this shows that the value updates are overestimated or underestimated indicating that in three-arcs-lookahead time the deterministic disruption status hardly stays the same. This causes lower solution quality. When we use double-pass, however, the expected travel time decreases as the update of the value function estimates are done if there is an improvement (Figure 4).

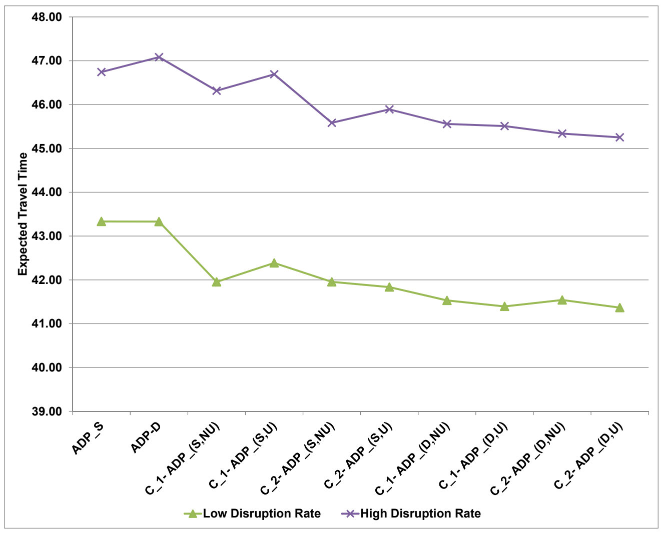
Path-Update versus No-Path-Update

In the hybrid algorithms, we either update the state values within the deterministic lookahead or not. Figures 3 and 4 show that the performance of the update of state values depends on whether we use double-pass or single-pass approach. In most of the network settings, single-pass algorithms with path-update perform worse than the single-pass no-path-update algorithms. In all of the networks, $C_1 - ADP_{(S,U)}$ performs significantly worse than C_1 and C_2 single-pass no-path-update algorithms. This is because in

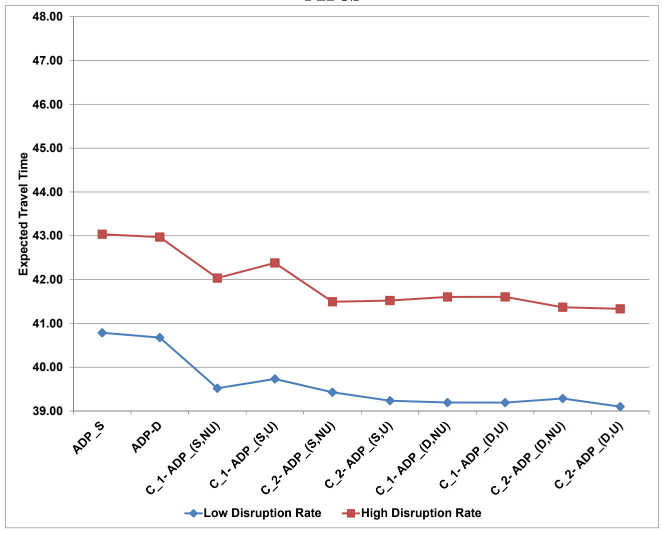
Figure 4: The expected travel time from ADP algorithms for low and high number of vulnerable arcs with 2 and 3 disruption levels



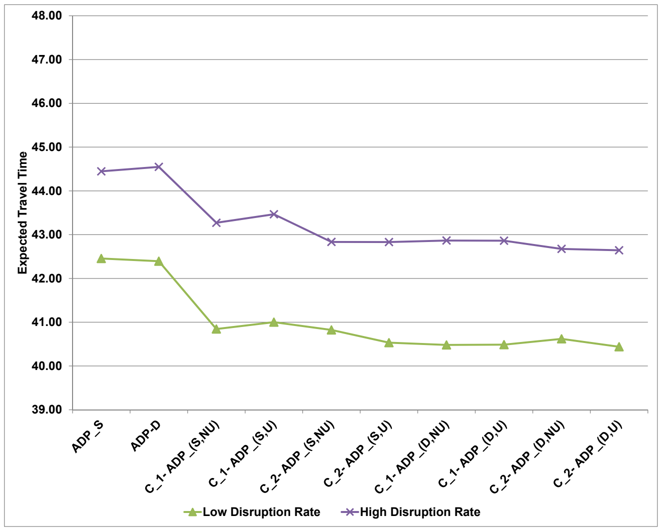
2 Disruption Levels- Lower Number of Vulnerable Arcs



2 Disruption Levels- Higher Number of Vulnerable Arcs



3 Disruption Levels- Lower Number of Vulnerable Arcs



3 Disruption Levels- Higher Number of Vulnerable Arcs

single-pass, we update all the state values in the path, no matter the update improves the solution or not. Furthermore, in the C_1 algorithms, due to the longer horizon of the deterministic lookahead policy, the value functions are more biased.

On the contrary, in the double-pass approach, the update procedure leads to higher solution quality. The double-pass no-path-update algorithms perform worse than the double-pass update algorithms. This is relevant with intuition as in the double-pass approach, we update the states values in the path, only if they improve the solution.

For the rest of the numerical analysis, we adopt the double-pass and the path-update properties for the hybrid ADP algorithms with clustering approach as the algorithms with these properties give consistently higher solution quality. For the standard ADP algorithms we adopt both the single-pass and the double-pass property.

Computational Time

Figure 5 shows the normalized computational times (in logarithmic scale) of the standard ADP algorithm with double pass, the hybrid ADP algorithm with clustering type 2 adopting double-pass and path-update approach and the benchmark heuristic $DP(2, H)$ with respect to the number of disruption levels. We observe that the hybrid ADP algorithm with the clustering approach's computational time has the lowest rate of increase as the state space increases exponentially. The standard ADP algorithm are much slower than the hybrid ADP algorithms. Due to the use of deterministic lookahead policy with the clustering approach, the updates of the values functions are faster with an efficient exploration/exploitation strategy. The computational time of $DP(2, H)$ increases at a higher rate when the number of disruption levels increases. Although $DP(2, H)$ has relatively good solution quality, for large scale networks with many disruption levels the hybrid ADP algorithms with the clustering approach becomes more attractive with lower computational time and high solution quality for practical implementation.

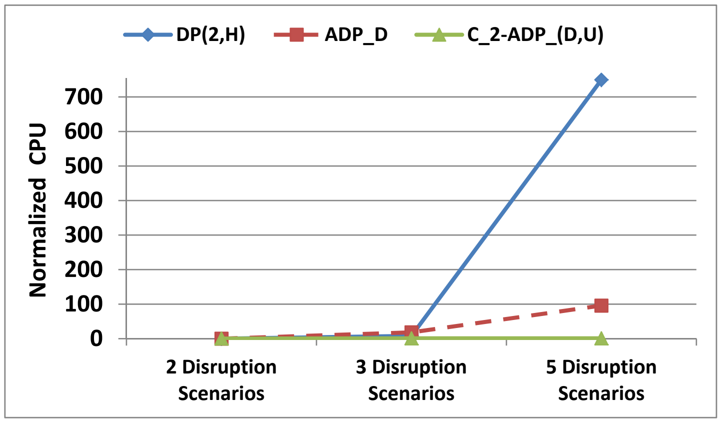


Figure 5: Relative Computation times for the different algorithms

Algorithm comparison: Solution quality

In this section, we analyze the performance of the ADP algorithms, the optimal algorithm (if applicable) and $DP(2, H)$ algorithm with respect to their total expected travel time. We also provide the percentage performance gap of each algorithm with respect to the benchmark algorithm, $DP(2, H)$. As we can not compute the optimal solution for all networks, we choose $DP(2, H)$ as the benchmark heuristic. We aggregate the average results over all 1800 network instances depending on the network size, the disruption rate and the rate of vulnerability. For each of these network properties, we investigate the performance of the algorithms on the networks with different disruption rates and various number of disruption levels.

Network Size

Tables 2, 3 and 4 show the expected travel time and the percentage gap with respect to $DP(2, H)$ with 2, 3 and 5 levels of disruptions respectively for different network sizes (Negative sign for the percentage gap means that the algorithm performs better than the benchmark heuristic).

In all of the network sizes, the hybrid ADP algorithms outperform the standard ADP algorithms. For instance, the performance gap difference between ADP_D and $C_2 - ADP_{(D,U)}$ increases from 3.31% (small size networks) to 5.28% (large size networks) for 2 disruption levels and low disruption probability (Table

2). Furthermore, as network size increases from 16 nodes to 64 nodes, the performance of the standard ADP algorithms decreases whereas the the performance of the hybrid ADP algorithms improves. This shows that the combination of deterministic lookahead policy with the value function approximation provides better updates of the value functions for larger networks. This performance change is more significant when the number of disruption states increases with the level of disruptions.

The experimental results show that on large size networks, the hybrid ADP algorithms with the clustering approach outperforms $DP(2, H)$ and the standard ADP algorithms regardless the disruption rate and the number of disruption levels. For instance, the average percentage performance gap of $C_2 - ADP_{(D,U)}$ changes from 0.39% (small size networks) to -0.29% (large size networks) with respect to $DP(2, H)$ (Tables 2 and 4). Also, we observe that the percentage gap between the optimal and the hybrid ADP algorithms decreases as the network size increases. This indicates the power of using a combination of a deterministic lookahead policy with value function approximations. When we proceed in clusters, the critical value functions are learned and these value functions are visited and updated more frequently. The rest of the value functions in the lookup table are handled within the deterministic lookahead policy. As a result of this, we obtain higher quality solutions with lower computation times (Tables 2, 3 and 4).

When we compare C_1 and C_2, we observe that C_2 performs on average similar or better than C_1 for all network sizes. As the number of disruption levels increases and the disruption rate becomes higher, C_2 outperforms C_1. When we consider, small size networks with low disruption probability, the performance gap difference between the $C_1 - ADP_{D,U}$ and $C_2 - ADP_{D,U}$ increases from 0.03%, 0.23% to 0.58% with 2, 3 and 5 disruption levels respectively (Tables 2, 3 and 4). This indicates that considering longer horizon for deterministic lookahead policy under high disruption rates and many disruption levels decreases the performance of the hybrid ADP algorithms. Thus, considering clusters of two-arcs-ahead neighborhood provides higher quality solutions than considering three-arcs-ahead neighborhood.

Vulnerability

Tables 5, 6 and 7 show the expected travel time and the percentage gap with respect to $DP(2, H)$ with 2, 3 and 5 levels of disruptions respectively for low and high vulnerability networks where the number of vulnerable arcs is either low or high.

Table 2: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with 16, 36 and 64 nodes, with 2 levels of disruptions

Network Size	Small Size Network				Medium Size Network				Large Size Network			
Disruption Rate	Low		High		Low		High		Low		High	
Algorithms\Performance	ETT*	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)
Opt	25.29	-2.60	29.48	-0.27	39.85	-0.21	43.97	-0.14	n.a	n.a	n.a	n.a
DP(2,H)	25.97	n.a	29.56	n.a	39.93	n.a	44.03	n.a	55.90	n.a	58.48	n.a
ADPS	26.96	3.82	30.02	1.55	41.82	4.74	44.96	2.11	58.11	3.96	60.52	3.48
ADPD	26.93	3.70	29.76	0.68	41.70	4.42	45.35	3.00	58.71	5.02	61.41	5.01
Cluster1- ADP(D,U)	26.08	0.42	29.66	0.33	40.15	0.55	44.41	0.86	55.94	0.07	58.71	0.40
Cluster2- ADP(D,U)	26.07	0.39	29.54	-0.05	40.12	0.47	44.24	0.49	55.76	-0.26	58.21	-0.46

* Expected Travel Time

Table 3: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with 16, 36 and 64 nodes, with 3 levels of disruptions

Network Size	Small Size Network				Medium Size Network				Large Size Network			
Disruption Rate	Low		High		Low		High		Low		High	
Algorithms\Performance	ETT*	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)
$DP(2, H)$	25.58	n.a.	27.99	n.a	39.95	n.a.	42.14	n.a	53.72	n.a.	55.98	n.a.
ADP_S	26.61	4.03	28.47	1.74	42.59	6.62	44.04	4.52	56.71	5.57	58.71	4.87
ADP_D	26.18	2.34	28.25	0.94	41.50	3.88	43.65	3.59	56.93	5.97	59.01	5.41
$C_1 - ADP_{(D,U)}$	25.74	0.63	28.08	0.34	40.25	0.77	42.53	0.94	53.52	-0.36	56.08	0.18
$C_2 - ADP_{(D,U)}$	25.68	0.40	27.93	-0.22	40.09	0.36	42.24	0.24	53.53	-0.34	55.80	-0.32

* Expected Travel Time

Table 4: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with 16, 36 and 64 nodes, with 5 levels of disruptions

Network Size	Small Size Network				Medium Size Network				Large Size Network			
Disruption Rate	Low		High		Low		High		Low		High	
Algorithms\Performance	ETT*	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)	ETT	Gap(%)
DP(2,H)	27.66	n.a.	29.21	n.a	40.93	n.a.	43.25	n.a	53.83	n.a.	57.40	n.a
ADPS	28.12	1.66	29.99	2.64	42.84	4.65	45.21	4.54	56.67	5.27	59.53	3.71
ADPD	27.92	0.94	29.36	0.50	42.92	4.86	44.88	3.77	57.33	6.49	60.40	5.23
Cluster1- ADP(D,U)	27.75	0.32	29.21	0.00	41.17	0.58	43.55	0.68	54.10	0.49	57.47	0.12
Cluster2- ADP(D,U)	27.58	-0.27	29.03	-0.62	41.05	0.28	43.27	0.04	53.68	-0.29	57.30	-0.16

* Expected Travel Time

According to the Tables 5, 6 and 7, the hybrid ADP algorithm always perform better than the standard ADP algorithm on both low and high vulnerable networks regardless the disruption rate and the number of disruption levels. As the networks become more vulnerable, the standard ADP algorithm performs significantly worse. For low vulnerable networks with 2 disruption levels, ADP_S performs on average 3.39% worse than the benchmark heuristic. As the network becomes highly vulnerable, this gap increases to 4.95%.

The hybrid ADP algorithms perform slightly worse than the optimal solution on low vulnerable networks with two disruption levels. Table 5 shows that the optimal algorithm performs 0.47% and 0.30% better than $DP(2, H)$ on the networks with low and high disruption rates respectively while $C_2 - ADP_{(D,U)}$ performs only 0.04% and 0.15% worse than $DP(2, H)$.

The numerical results show that as the number of disruption levels increases and the networks become highly vulnerable, the hybrid ADP algorithms perform better. For instance, the performance gap difference of $C_2 - ADP_{(D,U)}$ decreases from 0.19%, 0.08% and -0.13% for 2, 3 and 5 disruption levels respectively with high vulnerable networks under low disruption rate. Especially, C_2 algorithm performs better than $DP(2, H)$ on the high vulnerable networks with high disruption levels (Tables 5 versus 7).

Similar to the results regarding the network size, C_2 algorithm gives higher quality solutions then C_1 algorithm for both low and high vulnerable networks as C_1 considers the deterministic lookahead policy for a longer horizon. The results also show that as the networks become more vulnerable, the performance of C_1 improves.

7. Conclusion

In this paper, we consider dynamic shortest path problems with stochastic disruptions on a subset of arcs. Both historical and real-time information for the network are used for dynamic routing decisions. The disruption state of the following stages are dependent on the travel time of the current arc. We denote this property as travel-time-dependency. We model the problem as a discrete time finite horizon Markov Decision Process (MDP).

Table 5: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with low and high number of vulnerable arcs and 2 levels of disruptions

Vulnerability	Low Vulnerability				High Vulnerability			
Disruption Rate	Low		High		Low		High	
Algorithms\Performance	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)
Opt	39.72	-0.47	42.56	-0.30	n.a	n.a	n.a	n.a
DP(2,H)	39.91	n.a	42.68	n.a	41.29	n.a	45.36	n.a
ADPS	41.26	3.39	43.59	2.12	43.33	4.95	46.74	3.04
ADPD	41.56	4.14	43.93	2.93	43.33	4.94	47.08	3.79
Cluster1- ADP(D,U)	40.05	0.35	43.01	0.77	41.39	0.25	45.51	0.32
Cluster2- ADP(D,U)	39.93	0.04	42.75	0.15	41.37	0.19	45.25	-0.25

Table 6: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with low and high number of vulnerable arcs and 3 levels of disruptions

Vulnerability	Low Vulnerability				High Vulnerability			
Disruption Rate	Low		High		Low		High	
Algorithms\Performance	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)
$DP(2, H)$	39.09	n.a.	41.32	n.a.	40.40	n.a.	42.75	n.a.
ADP_S	41.15	5.27	43.04	4.14	42.79	5.90	44.45	3.97
ADP_D	40.68	4.05	42.72	3.39	42.39	4.92	44.55	4.21
$C_1 - ADP_{(D,U)}$	39.19	0.25	41.61	0.68	40.49	0.20	42.86	0.27
$C_2 - ADP_{(D,U)}$	39.10	0.02	41.33	0.02	40.44	0.08	42.64	-0.24

Table 7: The expected travel time and percentage gap with respect to $DP(2, H)$ for the routing algorithms in networks with low and high number of vulnerable arcs and 5 levels of disruptions

Vulnerability	Low Vulnerability				High Vulnerability			
Disruption Rate	Low		High		Low		High	
Algorithms\Performance	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)	Expected Value	Gap(%)
DP(2,H)	40.29	n.a.	42.34	n.a.	41.33	n.a.	44.23	n.a.
ADPS	42.40	5.24	44.52	5.14	42.68	3.27	45.30	2.41
ADPD	42.60	5.74	44.40	4.85	42.85	3.66	45.36	2.56
Cluster1- ADP(D,U)	40.46	0.43	42.50	0.37	41.55	0.53	44.32	0.19
Cluster2- ADP(D,U)	40.26	-0.06	42.29	-0.12	41.28	-0.13	44.11	-0.27

MDP formulation provides a practical framework to find dynamic routing decisions for each decision epoch. However, for large scale networks with many levels of disruptions, obtaining the optimal solution faces the curses of dimensionality, i.e., states, outcome, and decisions. Therefore, we use an Approximate Dynamic Programming (ADP) algorithm which is a well-known approximation approach. The ADP algorithms used in this paper are based on the value function approximations using a lookup table representation. First, we employ a standard value function approximation algorithm with various algorithmic design variations for updating the state values with efficient exploration/ exploitation strategies. Then, we improve the value function approximation algorithm by developing a hybrid ADP with a deterministic lookahead policy and value function approximations using a clustering approach. We develop two types of hybrid ADP algorithms. Type 1 considers a longer horizon (three-arcs-ahead neighborhood) for the deterministic lookahead policy and type 2 considers a shorter horizon (two-arcs-ahead neighborhood).

We develop a test bed of networks to evaluate the efficiency (both in computational time and solution quality) of our algorithms. The generated networks vary in network size, number of vulnerable arcs, number of disruption levels and rate of disruption for the vulnerable arc. We use a benchmark heuristic, a stochastic limited lookahead policy, shown to perform well in binary disruption levels. In our numerical analysis, we show that the hybrid ADP algorithms with the clustering approach outperforms the standard ADP algorithms. Furthermore, we observe that considering a shorter horizon for the deterministic lookahead policy improves the solution quality. The hybrid ADP algorithms outperforms the benchmark heuristic as the network size gets larger and the disruption rate gets higher.

The computational time of the hybrid ADP algorithms shows the slowest rate of increase with respect to the exponential increases in the state space. The computational time of the benchmark heuristic increases at a higher rate when the number of disruption levels increases. Although the benchmark heuristic has relatively good solution quality, for large scale networks with many disruption levels the hybrid ADP algorithms with the clustering approach becomes more attractive with lower computational time and high solution quality for practice.

This paper provides an exploratory analysis on solving the dynamic shortest path problems using hybrid ADP algorithms in large scale networks with many levels of disruptions. Furthermore, we provide an extensive analysis on the effect of using different algorithmic properties for ADP algorithms for the problem on hand. Future research involves integrating the hybrid ADP algorithms for the dynamic shortest path problems into the stochastic vehicle routing problem with dynamic travel times.

References

Barry, J. L., Kaelbling, L. P., Lozano-Pérez, T., 2011. Deth: approximate hierarchical solution of large markov decision processes. In: Proceedings of the twenty-second international joint conference on artificial intelligence. Vol. 3. pp. 1928–1935.

Bertsekas, D. P., Yu, H., 2010. Q-learning and enhanced policy iteration in discounted dynamic programming. Tech. rep.

Boutilier, C., Dearden, R., Goldszmidt, M., 2000. Stochastic dynamic programming with factored representations. Artificial Intelligence 121 (12), 49–107.

Dijkstra, E. W., 1959. A note on two problems in connexion with graphs. Numerische mathematik 1 (1), 269–271.

Fu, L., 2001. An adaptive routing algorithm for in-vehicle route guidance systems with real-time information. Transportation Research Part B: Methodological 35 (8), 749–765.

Givan, R., Dean, T., Greig, M., 2003. Equivalence notions and model minimization in markov decision processes. Artificial Intelligence 147 (1), 163–223.

Güner, A. R., Murat, A., Chinnam, R. B., 2012. Dynamic routing under recurrent and non-recurrent congestion using real-time its information. Computers and Operations Research 39 (2), 358–373.

Helbing, D., Treiber, M., Kesting, A., Schnhof, M., 2009. Theoretical vs. empirical classification and prediction of congested traffic states. The European Physical Journal B 69, 583–598.

Kim, K.-E., Dean, T., 2002. Solving factored mdps with large action space using algebraic decision diagrams. PRICAI 2002: Trends in Artificial Intelligence, 47–56.

Kim, S., Lewis, M. E., White, C. C. I., 2005a. State space reduction for nonstationary stochastic shortest path problems with real-time traffic information. IEEE Transactions on Intelligent Transportation Systems 6 (3), 273–284.

Kim, S., Lewis, M. E., White, C. C., I., 2005b. Optimal vehicle routing with real-time traffic information. IEEE Transactions on Intelligent Transportation Systems 6 (2), 178–188.

Polychronopoulos, G. H., Tsitsiklis, J. N., 1996. Stochastic shortest path problems with recourse. Networks 27 (2), 133–143.

Powell, W. B., 2007. Approximate dynamic programming: Solving the curses of dimensionality. John Wiley and Sons, Inc., New York.

Powell, W. B., 2011. Approximate dynamic programming: Solving the curses of dimensionality. John Wiley and Sons, Inc., New York.

Powell, W. B., Shapiro, J. A., Simo, H. P., 2002. An adaptive dynamic programming algorithm for the heterogeneous resource allocation problem. Transportation Science 36 (2), 231–249.

Powell, W. B., Simao, H. P., Bouzaiene-Ayari, B., 2012. Approximate dynamic programming in transportation and logistics: a unified framework. EURO Journal of Transportation and Logistics 1 (3), 237–284.

Powell, W. B., Van Roy, B., 2004. Approximate dynamic programming for high dimensional resource allocation problems. Handbook of learning and approximate dynamic programming, 261–280.

Rehborn, H., Klenov, S. L., Palmer, J., 2011. Common traffic congestion features studied in usa, uk, and germany based on kerner's three-phase traffic theory. In: Intelligent Vehicles Symposium (IV). IEEE, pp. 19–24.

Sever, D., Dellaert, N., van Woensel, T., de Kok, T., 2013. Dynamic shortest path problems: Hybrid routing policies considering network disruptions. Computers & Operations Research.

Simão, H. P., Day, J., George, A. P., Gifford, T., Nienow, J., Powell, W. B., 2009. An approximate dynamic programming algorithm for large-scale fleet management: A case application. Transportation Science 43 (2), 178–197.

Thomas, B., White, C. C., I., 2007. The dynamic shortest path problem with anticipation. European Journal of Operational Research 176 (2), 836–854.

Working Papers Beta 2009 - 2013

nr.	Year	Title	Author(s)
425	2013	Single Vehicle Routing with Stochastic Demands: Approximate Dynamic Programming	C. Zhang, N.P. Dellaert, L. Zhao, T. Van Woensel, D. Sever
424	2013	Influence of Spillback Effect on Dynamic Shortest Path Problems with Travel-Time-Dependent Network Disruptions	Derya Sever, Nico Dellaert, Tom Van Woensel, Ton de Kok
423	2013	Dynamic Shortest Path Problem with Travel-Time-Dependent Stochastic Disruptions: Hybrid Approximate Dynamic Programming Algorithms with a Clustering Approach	Derya Sever, Lei Zhao, Nico Dellaert, Tom Van Woensel, Ton de Kok
422	2013	System-oriented inventory models for spare parts	R.J.I. Basten, G.J. van Houtum
421	2013	Lost Sales Inventory Models with Batch Ordering And Handling Costs	T. Van Woensel, N. Erkip, A. Curseu, J.C. Fransoo
420	2013	Response speed and the bullwhip	Maximiliano Udenio, Jan C. Fransoo, Eleni Vatamidou, Nico Dellaert
419	2013	Anticipatory Routing of Police Helicopters	Rick van Urk, Martijn R.K. Mes, Erwin W. Hans
418	2013	Supply Chain Finance. A conceptual framework to advance research	Kasper van der Vliet, Matthew J. Reindorp, Jan C. Fransoo
417	2013	Improving the Performance of Sorter Systems By Scheduling Inbound Containers	S.W.A. Haneyah, J.M.J. Schutten, K. Fikse
416	2013	Regional logistics land allocation policies: Stimulating spatial concentration of logistics firms	Frank P. van den Heuvel, Peter W. de Langen, Karel H. van Donselaar, Jan C. Fransoo
415	2013	The development of measures of process harmonization	Heidi L. Romero, Remco M. Dijkman, Paul W.P.J. Grefen, Arjan van Weele
414	2013	*BASE/X.* Business Agility through Cross-Organizational Service Engineering	Paul Grefen, Egon Lüftenegger, Eric van der Linden, Caren Weisleder

413	2013	The Time-Dependent Vehicle Routing Problem with Soft Time Windows and Stochastic Travel Times	Duygu Tas, Nico Dellaert, Tom van Woensel, Ton de Kok
412	2013	Clearing the Sky - Understanding SLA Elements in Cloud Computing	Marco Comuzzi, Guus Jacobs, Paul Grefen
411	2013	Approximations for the waiting time distribution In an *M/G/c* priority queue	A. Al Hanbali, E.M. Alvarez, M.C. van der van der Heijden
410	2013	To co-locate or not? Location decisions and logistics concentration areas	Frank P. van den Heuvel, Karel H. van Donselaar, Rob A.C.M. Broekmeulen, Jan C. Fransoo, Peter W. de Langen
409	2013	The Time-Dependent Pollution-Routing Problem	Anna Franceschetti, Dorothée Honhon,Tom van Woensel, Tolga Bektas, GilbertLaporte.
408	2013	Scheduling the scheduling task: A time Management perspective on scheduling	J.A. Larco, V. Wiers, J. Fransoo
407	2013	Clustering Clinical Departments for Wards to Achieve a Prespecified Blocking Probability	J. Theresia van Essen, Mark van Houdenhoven, Johann L. Hurink
406	2013	MyPHRMachines: Personal Health Desktops in the Cloud	Pieter Van Gorp, Marco Comuzzi
405	2013	Maximising the Value of Supply Chain Finance	Kasper van der Vliet, Matthew J. Reindorp, Jan C. Fransoo
404	2013	Reaching 50 million nanostores: retail distribution in emerging megacities	Edgar E. Blanco, Jan C. Fransoo
403	2013	A Vehicle Routing Problem with Flexible Time Windows	Duygu Tas, Ola Jabali, Tom van Woensel
402	2013	The Service Dominant Business Model: A Service Focused Conceptualization	Egon Lüftenegger, Marco Comuzzi, Paul Grefen, Caren Weisleder
401	2012	Relationship between freight accessibility and Logistics employment in US counties	Frank P. van den Heuvel, Liliana Rivera,Karel H. van Donselaar, Ad de Jong,Yossi Sheffi, Peter W. de Langen, Jan C.Fransoo

400	2012	A Condition-Based Maintenance Policy for Multi-Component Systems with a High Maintenance Setup Cost	Qiushi Zhu, Hao Peng, Geert-Jan van Houtum
399	2012	A flexible iterative improvement heuristic to Support creation of feasible shift rosters in Self-rostering	E. van der Veen, J.L. Hurink, J.M.J. Schutten, S.T. Uijland
398	2012	Scheduled Service Network Design with Synchronization and Transshipment Constraints For Intermodal Container Transportation Networks	K. Sharypova, T.G. Crainic, T. van Woensel, J.C. Fransoo
397	2012	Destocking, the bullwhip effect, and the credit Crisis: empirical modeling of supply chain Dynamics	Maximiliano Udenio, Jan C. Fransoo, Robert Peels
396	2012	Vehicle routing with restricted loading capacities	J. Gromicho, J.J. van Hoorn, A.L. Kok J.M.J. Schutten
395	2012	Service differentiation through selective lateral transshipments	E.M. Alvarez, M.C. van der Heijden, I.M.H. Vliegen, W.H.M. Zijm
394	2012	A Generalized Simulation Model of an Integrated Emergency Post	Martijn Mes, Manon Bruens
393	2012	Business Process Technology and the Cloud: Defining a Business Process Cloud Platform	Vasil Stoitsev, Paul Grefen
392	2012	Vehicle Routing with Soft Time Windows and Stochastic Travel Times: A Column Generation And Branch-and-Price Solution Approach	D. Tas, M. Gendreau, N. Dellaert, T. van Woensel, A.G. de Kok
391	2012	Improve OR-Schedule to Reduce Number of Required Beds	J.T. v. Essen, J.M. Bosch, E.W. Hans, M. v. Houdenhoven, J.L. Hurink
390	2012	How does development lead time affect performance over the ramp-up lifecycle?	Andres Pufall, Jan C. Fransoo, Ad de Jong
389	2012	Evidence from the consumer electronics industry	Andreas Pufall, Jan C. Fransoo, Ad de Jong, Ton de Kok

388	2012	The Impact of Product Complexity on Ramp-Up Performance	Frank P.v.d. Heuvel, Peter W.de Langen, Karel H. v. Donselaar, Jan C. Fransoo
387	2012	Co-location synergies: specialized versus diverse logistics concentration areas	Frank P.v.d. Heuvel, Peter W.de Langen, Karel H. v.Donselaar, Jan C. Fransoo
386	2012	Proximity matters: Synergies through co-location of logistics establishments	Frank P. v.d.Heuvel, Peter W.de Langen, Karel H.v. Donselaar, Jan C. Fransoo
385	2012	Spatial concentration and location dynamics in logistics:the case of a Dutch province	Zhiqiang Yan, Remco Dijkman, Paul Grefen
384	2012	FNet: An Index for Advanced Business Process Querying	W.R. Dalinghaus, P.M.E. Van Gorp
383	2012	Defining Various Pathway Terms	Egon Lüftenegger, Paul Grefen, Caren Weisleder
382	2012	The Service Dominant Strategy Canvas: Defining and Visualizing a Service Dominant Strategy through the Traditional Strategic Lens	Stefano Fazi, Tom van Woensel, Jan C. Fransoo
381	2012	A Stochastic Variable Size Bin Packing Problem With Time Constraints	K. Sharypova, T. van Woensel, J.C. Fransoo
380	2012	Coordination and Analysis of Barge Container Hinterland Networks	Frank P. van den Heuvel, Peter W. de Langen, Karel H. van Donselaar, Jan C. Fransoo
379	2012	Proximity matters: Synergies through co-location of logistics establishments	Heidi Romero, Remco Dijkman, Paul Grefen, Arjan van Weele
378	2012	A literature review in process harmonization: a conceptual framework	S.W.A. Haneya, J.M.J. Schutten, P.C. Schuur, W.H.M. Zijm
377	2012	A Generic Material Flow Control Model for Two Different Industries	H.G.H. Tiemessen, M. Fleischmann, G.J. van Houtum, J.A.E.E. van Nunen, E. Pratsini
375	2012	Improving the performance of sorter systems by scheduling inbound containers	Albert Douma, Martijn Mes

374	2012	Strategies for dynamic appointment making by container terminals	Pieter van Gorp, Marco Comuzzi
373	2012	MyPHRMachines: Lifelong Personal Health Records in the Cloud	E.M. Alvarez, M.C. van der Heijden, W.H.M. Zijm
372	2012	Service differentiation in spare parts supply through dedicated stocks	Frank Karsten, Rob Basten
371	2012	Spare parts inventory pooling: how to share the benefits	X.Lin, R.J.I. Basten, A.A. Kranenburg, G.J. van Houtum
370	2012	Condition based spare parts supply	Martijn Mes
369	2012	Using Simulation to Assess the Opportunities of Dynamic Waste Collection	J. Arts, S.D. Flapper, K. Vernooij
368	2011	Aggregate overhaul and supply chain planning for rotables	J.T. van Essen, J.L. Hurink, W. Hartholt, B.J. van den Akker
367	2011	Operating Room Rescheduling	Kristel M.R. Hoen, Tarkan Tan, Jan C. Fransoo, Geert-Jan van Houtum
366	2011	Switching Transport Modes to Meet Voluntary Carbon Emission Targets	Elisa Alvarez, Matthieu van der Heijden
365	2011	On two-echelon inventory systems with Poisson demand and lost sales	J.T. van Essen, E.W. Hans, J.L. Hurink, A. Oversberg
364	2011	Minimizing the Waiting Time for Emergency Surgery	Duygu Tas, Nico Dellaert, Tom van Woensel, Ton de Kok
363	2011	Vehicle Routing Problem with Stochastic Travel Times Including Soft Time Windows and Service Costs	Erhun Özkan, Geert-Jan van Houtum, Yasemin Serin
362	2011	A New Approximate Evaluation Method for Two-Echelon Inventory Systems with Emergency Shipments	Said Dabia, El-Ghazali Talbi, Tom Van Woensel, Ton de Kok
361	2011	Approximating Multi-Objective Time-Dependent Optimization Problems	Said Dabia, Stefan Röpke, Tom Van Woensel, Ton de Kok

360	2011	Branch and Cut and Price for the Time Dependent Vehicle Routing Problem with Time Window	A.G. Karaarslan, G.P. Kiesmüller, A.G. de Kok
359	2011	Analysis of an Assemble-to-Order System with Different Review Periods	Ahmad Al Hanbali, Matthieu van der Heijden
358	2011	Interval Availability Analysis of a Two-Echelon, Multi-Item System	Felipe Caro, Charles J. Corbett, Tarkan Tan, Rob Zuidwijk
357	2011	Carbon-Optimal and Carbon-Neutral Supply Chains	Sameh Haneyah, Henk Zijm, Marco Schutten, Peter Schuur
356	2011	Generic Planning and Control of Automated Material Handling Systems: Practical Requirements Versus Existing Theory	M. van der Heijden, B. Iskandar
355	2011	Last time buy decisions for products sold under warranty	Frank P. van den Heuvel, Peter W. de Langen, Karel H. van Donselaar, Jan C. Fransoo
354	2011	Spatial concentration and location dynamics in logistics: the case of a Dutch provence	Frank P. van den Heuvel, Peter W. de Langen, Karel H. van Donselaar, Jan C. Fransoo
353	2011	Identification of Employment Concentration Areas	Pieter van Gorp, Remco Dijkman
352	2011	BOMN 2.0 Execution Semantics Formalized as Graph Rewrite Rules: extended version	Frank Karsten, Marco Slikker, Geert-Jan van Houtum
351	2011	Resource pooling and cost allocation among independent service providers	E. Lüftenegger, S. Angelov, P. Grefen
350	2011	A Framework for Business Innovation Directions	Remco Dijkman, Irene Vanderfeesten, Hajo A. Reijers
349	2011	The Road to a Business Process Architecture: An Overview of Approaches and their Use	K.M.R. Hoen, T. Tan, J.C. Fransoo G.J. van Houtum
348	2011	Effect of carbon emission regulations on transport mode selection under stochastic demand	Murat Firat, Cor Hurkens
347	2011	An improved MIP-based combinatorial approach for a multi-skill workforce scheduling problem	R.J.I. Basten, M.C. van der Heijden, J.M.J. Schutten
346	2011	An approximate approach for the joint problem of level of repair analysis and spare parts stocking	R.J.I. Basten, M.C. van der Heijden, J.M.J. Schutten
		Joint optimization of level of repair analysis and spare parts stocks	

345	2011	Inventory control with manufacturing lead time flexibility	Ton G. de Kok
344	2011	Analysis of resource pooling games via a new extenstion of the Erlang loss function	Frank Karsten, Marco Slikker, Geert-Jan van Houtum
343	2011	Vehicle refueling with limited resources	Murat Firat, C.A.J. Hurkens, Gerhard J. Woeginger
342	2011	Optimal Inventory Policies with Non-stationary Supply Disruptions and Advance Supply Information	Bilge Atasoy, Refik Güllü, TarkanTan
341	2010	Redundancy Optimization for Critical Components in High-Availability Capital Goods	Kurtulus Baris Öner, Alan Scheller-Wolf Geert-Jan van Houtum
339	2010	Analysis of a two-echelon inventory system with two supply modes	Joachim Arts, Gudrun Kiesmüller
338	2010	Analysis of the dial-a-ride problem of Hunsaker and Savelsbergh	Murat Firat, Gerhard J. Woeginger
335	2010	Attaining stability in multi-skill workforce scheduling	Murat Firat, Cor Hurkens
334	2010	Flexible Heuristics Miner (FHM)	A.J.M.M. Weijters, J.T.S. Ribeiro
333	2010	An exact approach for relating recovering surgical patient workload to the master surgical schedule	P.T. Vanberkel, R.J. Boucherie, E.W. Hans, J.L. Hurink, W.A.M. van Lent, W.H. van Harten
332	2010	Efficiency evaluation for pooling resources in health care	Peter T. Vanberkel, Richard J. Boucherie, Erwin W. Hans, Johann L. Hurink, Nelly Litvak
331	2010	The Effect of Workload Constraints in Mathematical Programming Models for Production Planning	M.M. Jansen, A.G. de Kok, I.J.B.F. Adan
330	2010	Using pipeline information in a multi-echelon spare parts inventory system	Christian Howard, Ingrid Reijnen, Johan Marklund, Tarkan Tan
329	2010	Reducing costs of repairable spare parts supply systems via dynamic scheduling	H.G.H. Tiemessen, G.J. van Houtum

328	2010	Identification of Employment Concentration and Specialization Areas: Theory and Application	F.P. van den Heuvel, P.W. de Langen, K.H. van Donselaar, J.C. Fransoo
327	2010	A combinatorial approach to multi-skill workforce scheduling	Murat Firat, Cor Hurkens
326	2010	Stability in multi-skill workforce scheduling	Murat Firat, Cor Hurkens, Alexandre Laugier
325	2010	Maintenance spare parts planning and control: A framework for control and agenda for future research	M.A. Driessen, J.J. Arts, G.J. v. Houtum, W.D. Rustenburg, B. Huisman
324	2010	Near-optimal heuristics to set base stock levels in a two-echelon distribution network	R.J.I. Basten, G.J. van Houtum
323	2010	Inventory reduction in spare part networks by selective throughput time reduction	M.C. van der Heijden, E.M. Alvarez, J.M.J. Schutten
322	2010	The selective use of emergency shipments for service-contract differentiation	E.M. Alvarez, M.C. van der Heijden, W.H. Zijm
321	2010	Heuristics for Multi-Item Two-Echelon Spare Parts Inventory Control Problem with Batch Ordering in the Central Warehouse	B. Walrave, K. v. Oorschot, A.G.L. Romme
320	2010	Preventing or escaping the suppression mechanism: intervention conditions	Nico Dellaert, Jully Jeunet.
319	2010	Hospital admission planning to optimize major resources utilization under uncertainty	R. Seguel, R. Eshuis, P. Grefen.
318	2010	Minimal Protocol Adaptors for Interacting Services	Tom Van Woensel, Marshall L. Fisher, Jan C. Fransoo.
317	2010	Teaching Retail Operations in Business and Engineering Schools	Lydie P.M. Smets, Geert-Jan van Houtum, Fred Langerak.
316	2010	Design for Availability: Creating Value for Manufacturers and Customers	Pieter van Gorp, Rik Eshuis.
		Transforming Process Models: executable rewrite	

		rules versus a formalized Java program	
315	2010	Getting trapped in the suppression of exploration: A simulation model	Bob Walrave, Kim E. van Oorschot, A. Georges L. Romme
314	2010	A Dynamic Programming Approach to Multi-Objective Time-Dependent Capacitated Single Vehicle Routing Problems with Time Windows	S. Dabia, T. van Woensel, A.G. de Kok
313	2010		
312	2010	Tales of a So(u)rcerer: Optimal Sourcing Decisions Under Alternative Capacitated Suppliers and General Cost Structures	Osman Alp, Tarkan Tan
311	2010	In-store replenishment procedures for perishable inventory in a retail environment with handling costs and storage constraints	R.A.C.M. Broekmeulen, C.H.M. Bakx
310	2010	The state of the art of innovation-driven business models in the financial services industry	E. Lüftenegger, S. Angelov, E. van der Linden, P. Grefen
309	2010	Design of Complex Architectures Using a Three Dimension Approach: the CrossWork Case	R. Seguel, P. Grefen, R. Eshuis
308	2010	Effect of carbon emission regulations on transport mode selection in supply chains	K.M.R. Hoen, T. Tan, J.C. Fransoo, G.J. van Houtum
307	2010	Interaction between intelligent agent strategies for real-time transportation planning	Martijn Mes, Matthieu van der Heijden, Peter Schuur
306	2010	Internal Slackening Scoring Methods	Marco Slikker, Peter Borm, René van den Brink
305	2010	Vehicle Routing with Traffic Congestion and Drivers' Driving and Working Rules	A.L. Kok, E.W. Hans, J.M.J. Schutten, W.H.M. Zijm
304	2010	Practical extensions to the level of repair analysis	R.J.I. Basten, M.C. van der Heijden, J.M.J. Schutten
303	2010	Ocean Container Transport: An Underestimated and Critical Link in Global Supply Chain Performance	Jan C. Fransoo, Chung-Yee Lee
302	2010	Capacity reservation and utilization for a manufacturer with uncertain capacity and demand	Y. Boulaksil; J.C. Fransoo; T. Tan
300	2009	Spare parts inventory pooling games	F.J.P. Karsten; M. Slikker; G.J. van Houtum
299	2009	Capacity flexibility allocation in an outsourced supply chain with reservation	Y. Boulaksil, M. Grunow, J.C. Fransoo
298	2010	An optimal approach for the joint problem of level of repair analysis and spare parts stocking	R.J.I. Basten, M.C. van der Heijden, J.M.J. Schutten
297	2009	Responding to the Lehman Wave: Sales Forecasting and Supply Management during the Credit Crisis	Robert Peels, Maximiliano Udenio, Jan C. Fransoo, Marcel Wolfs, Tom Hendrikx
296	2009	An exact approach for relating recovering surgical patient workload to the master surgical schedule	Peter T. Vanberkel, Richard J. Boucherie, Erwin W. Hans, Johann L. Hurink, Wineke A.M. van Lent, Wim H. van Harten

295	2009	An iterative method for the simultaneous optimization of repair decisions and spare parts stocks	R.J.I. Basten, M.C. van der Heijden, J.M.J. Schutten
294	2009	Fujaba hits the Wall(-e)	Pieter van Gorp, Ruben Jubeh, Bernhard Grusie, Anne Keller
293	2009	Implementation of a Healthcare Process in Four Different Workflow Systems	R.S. Mans, W.M.P. van der Aalst, N.C. Russell, P.J.M. Bakker
292	2009	Business Process Model Repositories - Framework and Survey	Zhiqiang Yan, Remco Dijkman, Paul Grefen
291	2009	Efficient Optimization of the Dual-Index Policy Using Markov Chains	Joachim Arts, Marcel van Vuuren, Gudrun Kiesmuller
290	2009	Hierarchical Knowledge-Gradient for Sequential Sampling	Martijn R.K. Mes; Warren B. Powell; Peter I. Frazier
289	2009	Analyzing combined vehicle routing and break scheduling from a distributed decision making perspective	C.M. Meyer; A.L. Kok; H. Kopfer; J.M.J. Schutten
288	2009	Anticipation of lead time performance in Supply Chain Operations Planning	Michiel Jansen; Ton G. de Kok; Jan C. Fransoo
287	2009	Inventory Models with Lateral Transshipments: A Review	Colin Paterson; Gudrun Kiesmuller; Ruud Teunter; Kevin Glazebrook
286	2009	Efficiency evaluation for pooling resources in health care	P.T. Vanberkel; R.J. Boucherie; E.W. Hans; J.L. Hurink; N. Litvak
285	2009	A Survey of Health Care Models that Encompass Multiple Departments	P.T. Vanberkel; R.J. Boucherie; E.W. Hans; J.L. Hurink; N. Litvak
284	2009	Supporting Process Control in Business Collaborations	S. Angelov; K. Vidyasankar; J. Vonk; P. Grefen
283	2009	Inventory Control with Partial Batch Ordering	O. Alp; W.T. Huh; T. Tan
282	2009	Translating Safe Petri Nets to Statecharts in a Structure-Preserving Way	R. Eshuis
281	2009	The link between product data model and process model	J.J.C.L. Vogelaar; H.A. Reijers
280	2009	Inventory planning for spare parts networks with delivery time requirements	I.C. Reijnen; T. Tan; G.J. van Houtum
279	2009	Co-Evolution of Demand and Supply under Competition	B. Vermeulen; A.G. de Kok
278	2010	Toward Meso-level Product-Market Network Indices for Strategic Product Selection and (Re)Design Guidelines over the Product Life-Cycle	B. Vermeulen, A.G. de Kok
277	2009	An Efficient Method to Construct Minimal Protocol Adaptors	R. Seguel, R. Eshuis, P. Grefen
276	2009	Coordinating Supply Chains: a Bilevel Programming Approach	Ton G. de Kok, Gabriella Muratore
275	2009	Inventory redistribution for fashion products under	G.P. Kiesmuller, S. Minner

		demand parameter update	
274	2009	Comparing Markov chains: Combining aggregation and precedence relations applied to sets of states	A. Busic, I.M.H. Vliegen, A. Scheller-Wolf
273	2009	Separate tools or tool kits: an exploratory study of engineers' preferences	I.M.H. Vliegen, P.A.M. Kleingeld, G.J. van Houtum
272	2009	An Exact Solution Procedure for Multi-Item Two-Echelon Spare Parts Inventory Control Problem with Batch Ordering	Engin Topan, Z. Pelin Bayindir, Tarkan Tan
271	2009	Distributed Decision Making in Combined Vehicle Routing and Break Scheduling	C.M. Meyer, H. Kopfer, A.L. Kok, M. Schutten
270	2009	Dynamic Programming Algorithm for the Vehicle Routing Problem with Time Windows and EC Social Legislation	A.L. Kok, C.M. Meyer, H. Kopfer, J.M.J. Schutten
269	2009	Similarity of Business Process Models: Metics and Evaluation	Remco Dijkman, Marlon Dumas, Boudewijn van Dongen, Reina Kaarik, Jan Mendling
267	2009	Vehicle routing under time-dependent travel times: the impact of congestion avoidance	A.L. Kok, E.W. Hans, J.M.J. Schutten
266	2009	Restricted dynamic programming: a flexible framework for solving realistic VRPs	J. Gromicho; J.J. van Hoorn; A.L. Kok; J.M.J. Schutten;

Working Papers published before 2009 see: http://beta.ieis.tue.nl