Dynamic Routing and Wavelength Assignment Using First Policy Iteration, Inhomogeneous Traffic Case

Esa Hyytiä and Jorma Virtamo

Helsinki University of Technology
Laboratory of Telecommunications Technology
P.O.Box 3000, FIN-02150 HUT, Finland
E-mail: {esa.hyytia, jorma.virtamo}@hut.fi

April 3, 2000

Abstract

The routing and wavelength assignment problem (RWA) in WDM network can be viewed as a Markov Decision Process (MDP). The problem, however, defies calculation of the exact solution because of the huge size of the state space. Several heuristic algorithms have been presented in the literature. Generally, these algorithms, however, do not take into account the available extra information about the traffic, e.g. inhomogeneous arrival rates. In this paper we propose an approach where, starting from a given heuristic algorithm, one obtains a better algorithm by the first policy iteration. At each decision epoch a decision analysis is made where the costs of all the alternative actions are estimated by simulations on the fly. Being computationally intensive, this method can be used in real time only for systems with slow dynamics. Off-line it can be used to assess how close the heuristic algorithms come to the optimal policy. Numerical examples are given about the policy improvement.

1 Introduction

The wavelength division multiplexing (WDM) is a promising technology for future all-optical networks. In WDM several optical signals using different wavelengths share same fibre. The capacity of such fibre links can be huge, even terabits per second. The routing in network nodes is based on wavelengths of incoming signals [1][2][3].

Generally, the routing and wavelength assignment (RWA) problem in WDM networks consists of choosing a route and a wavelength for each connection so that no two connections using the same wavelength share the same fibre [4][5]. For example a simple form of RWA problem is a static traffic case with single fibre links. If the nodes are incapable to do wavelength translations, an assumption made throughout this work, the problem can be mapped to a node coloring problem, once routing is fixed (see e.g. [4]).

When the traffic is not static, lightpath requests arrive randomly following some traffic process. Connection requests between a given source destination pair constitute a *traffic*

class, which we index by k, $k \in \mathcal{K}$, where $\mathcal{K}$ is the set of all source destination pairs. Usually the RWA algorithm configures lightpaths in the network unless there are not enough resources available and the request is blocked (see e.g. [6][7][8]).

The possible schemes considered under dynamical traffic can be divided into two cases. If it is possible to reconfigure the whole network when blocking would occur the blocking probability can be considerably reduced. Such an operation, however, interrupts all (or at least many) active lightpaths and requires a lot of coordination between all the nodes. In large networks the reconfiguration seems infeasible. In any case the reconfiguration algorithm should try to minize the number of reconfigured lightpaths [9].

The other case is when active lightpaths may not be reconfigured. In this case it is important to decide which route and wavelength are assigned to an arriving connection request in order to balance the load and minimize the future congestion in the network.

Several heuristic algorithms have been proposed and studied (see e.g. [6][7][8]). These algorithms however do not take into account the traffic characteristics like unequal costs of different requests or inhomogeneous arrival rates. In this paper we study this problem in the setting of Markov Decision Processes (MDP) and propose a approach, where we try to improve any given heuristic algorithm by the first policy iteration [10][11]. The policy iteration, indeed, is known to lead to a new policy with better performance. In order to avoid dealing with the huge size of the state space in calculating the relative state costs needed in the policy improvement step, we suggest to estimate these costs on the fly by simulations for the limited set of states that are relevant at any given decision epoch, i.e. when the route and wavelength assignment for an arriving call has to be made. This approach was introduced in [12]. In this paper we focus on the performance improvement in heterogeneous traffic environment, while the traffic characteristics in [12] were assumed to be uniform.

The rest of the paper is organized as follows. In section 2 we briefly review Markov Decision Processes and policy iteration in general, and the first policy iteration, in particular. In section 3, we consider the relative costs of states and how they are used in the policy iteration, and in the following section 4 we study how theses state costs can be estimated by simulations. Some heuristic RWA algorithms are presented in section 5. These are used as a starting point for policy iteration, and, in section 6 some numerical results obtained by simulations are presented. Finally, section 7 contains conclusions.

2 Policy Iteration

Routing and wavelength allocation constitute a typical decision making problem. When certain events occur, one has to decide on some action. In the RWA problem, in particular, upon arrival of a request for new connection one has to decide whether or not to accept the request, and if accepted, which resources to allocate for it, i.e. which of available routes and wavelengths are used for that connection.

In general, one is interested in the optimal policy which maximizes or minimizes the expectation (infinite time horizon) of a given objective function. Here we assume that the objective is defined in terms of minimizing some cost function. The cost may represent e.g. the loss of revenue due to blocked calls, where different revenue may be associated to each type of call.

When the arrival process of type k calls is a Poisson process with intensity λ_k, the holding times of those calls are distributed exponentially with mean $1/\mu_k$ and the expected revenue per carried call is w_k, then the system constitutes a Markov Process and the problem of determining the optimal policy belongs to the class of Markov Decision Processes (MDP) described e.g. in [10] and [11].

Three main approaches for solving the optimal policy in the MDP setting are the policy iteration, value iteration and linear programming approach. In this paper, we concentrate on the policy iteration, where, as the name says, one tries to find the optimal policy by starting from some policy and iteratively improving it. This policy iteration is known to converge rather quickly to the optimal policy. Even the first iteration often yields a new policy which is rather close to the optimal one. In practice, it is seldom possible to go beyond the first iteration. Also in this work, we will restrict ourselves to the first policy iteration.

At each decision epoch, i.e. arrival of a new request, there is a finite set of possible actions: either reject the call or accept it and assign a feasible combination of route and wavelength (RW) to it. A feasible RW combination is such that along the route from the source to destination the wavelength is not being used on any of the links. If no feasible RW combination exists, the call is unconditionally rejected.

A *policy* defines for each possible state of the system and for each class k of an arriving call which of the possible actions is taken. Many heuristic policies have been proposed in the literature such as the first-fit wavelength and most-used wavelength policies combined with shortest path routing or near shortest path routing [6][7][8]. Some of them work reasonably well. Common to all heuristic policies is that they are simple. The choice of the action to be taken at each decision epoch can usually be described in simple terms and does not require much computation. We take one of the heuristic policies as a starting point and call it the *standard policy*. The policy resulting from the first policy iteration we refer to as the *iteration policy*.

By doing the first policy iteration we have two goals in mind. 1) Finding a better RWA algorithm which, being computationally intensive, may or may not be calculable in real time, depending on the time scale of the dynamics of the system. 2) Even in the case the algorithm is not calculable in real time, estimating how far the performance of a heuristic algorithm is from the optimal one.

Briefly, as explained in more detail below, our idea in the policy iteration is the following: at each decision epoch we make a decision analysis of all the alternative actions. For each of the possible actions, i.e. decision alternatives, we estimate the future costs by simulation. Thus, assuming that a given action is taken we let the system proceed from the state where it is after that action and use the standard policy to make all the subsequent decisions in the simulation. The iteration policy is the policy which is obtained when at each decision epoch the action is chosen for which the estimated cost is the minimum. It can be shown that the iteration policy is always better or at least as good a policy as the standard policy, and as said, it often comes rather close to the optimal policy.

3 Relative costs of states

In the MDP theory, the first policy iteration consists of the following steps: With the standard policy one solves the Howard equations (see, e.g. [10][11]) to give the so called relative costs of the states, C_i, which for each possible state i of the system describe the difference in the expected cumulative cost from time 0 to infinity, given that the system starts from state i rather than from the equilibrium. Then, given that the current state of the system is j and a class-k call is offered, one calculates the cost $C_j + w_k$ for the action that the call is rejected, and the cost C_i, $i \in \mathcal{A}(j,k)$, for the case the call is accepted, where $\mathcal{A}(j,k)$ is the set of states reachable from state j by assigning call-k a feasible RW pair. By choosing always the action which minimizes the cost, one gets the iteration policy, i.e. the policy resulting from the first policy iteration.

Though the Howard equations are just a set of linear equations for relative costs C_i and the average cost rate c of the standard policy (see below), their solution cannot be obtained because of the prohibitive size of the state space of any realistic system. However, at any decision epoch the relative costs C_i are needed only for the current state j and a small set of states $\mathcal{A}(j,k)$ reachable from the current state. We propose to estimate these values on the fly by means of simulations. To this end, it is useful to consider the physical interpretation of the relative costs C_i.

Given that the system starts from state i at time 0 and standard policy is applied for all decisions, the cumulative costs are accrued at the expected rate $c_t(i)$ at time t,

$$c_t(i) = \sum \lambda_k w_k \mathrm{P}\{I_t \in \mathcal{B}_k | I_0 = i\}, \tag{1}$$

i.e. the expected rate of lost revenue, where $\mathrm{P}\{I_t \in \mathcal{B}_k\}$ is the probability that at time t the state I_t of the system is a blocking state for class-k calls. When $I_t \in \mathcal{B}_k$ class-k calls arriving at time t are blocked by the standard policy because either no feasible RW pair exists or the policy otherwise deems the blocking to be advantageous in the long run. The expected cost rate $c_t(i)$ depends on the initial state i. However, no matter what the initial state is, as t tends to infinity, the expected cost rate tends to a constant c, which is specific to the standard policy, and corresponds to (1) with steady state blocking probabilities $\mathrm{P}\{I_t \in \mathcal{B}_k\}$.

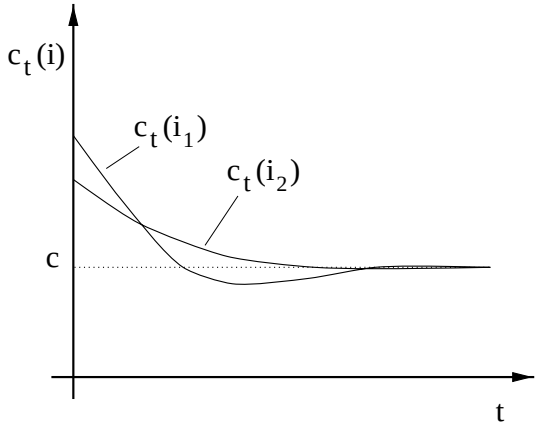

Figure 1: Expected costs with different initial choices as a function of time.

The behavior of the function $c_t(i)$ is depicted in Figure 1 for two different initial values i_1

and i_2. The relative cost C_i is defined as the integral

$$C_i = \int_0^\infty (c_t(i) - c)\, dt,$$

i.e. the area between the curve $c_t(i)$ and the line at level c. So we are interested in the transient behavior of $c_t(i)$; after the transient no contribution comes to integral. The length of the transient is of the order $1/\mu$, where $1/\mu$ is the maximum over $\{1/\mu_k\}$, $k \in \mathcal{K}$. After that the system essentially forgets the information about the initial state. So we can restrict ourselves to an appropriately chosen finite interval $(0, T)$. The actual choice of T is a tradeoff between different considerations as will be discussed later.

One easily sees that in the policy improvement step only the differences of the values C_i between different states are important. Therefore, we can neglect the constant c in the integral, as it is common to all states, and end up for thus redefined C_i,

$$C_i \approx C_i(T) = \int_0^T c_t(i)\, dt, \tag{2}$$

which is simply the expected cumulative cost in interval $(0, T)$ starting from the initial state i.

4 Estimation of the state costs by simulation

In practice, it is not feasible to calculate the cost rate function $c_t(i)$ analytically even for the simplest policies. Therefore, we estimate the state costs C_i by simulations. In each simulation the system is initially set in state i and then the evolution of the system is followed for the period of length T, making all the RWA decisions according to the standard policy.

4.1 Statistics collection: blocking time vs. blocking events

In collecting the statistics one has two alternatives. Either one records the time intervals when the system is in a blocking state of class-k calls, for all $k \in \mathcal{K}$. If the cumulative time within interval $(0, T)$ when the system is in the blocking state of class-k calls is denoted by $\tau_k(i)$, then the integral is simply

$$\hat{C}_i = \sum \lambda_k w_k \tau_k(i). \tag{3}$$

Alternatively, one records the number $\nu_k(i)$ of blocked calls of type k in interval $(0, T)$. Then we have

$$\hat{C}_i = \sum w_k \nu_k(i). \tag{4}$$

In these equations we have written explicitly $\tau_k(i)$ and $\nu_k(i)$ in order to emphasize that the system starts from the state i. Both (3) and (4) give an unbiased estimate for C_i. In either case, the simulation has to be repeated a number of times in order to get an estimator with small enough a confidence interval.

Denote the estimates of future costs obtained in the jth simulation run by $\hat{C}_i^{(j)}$, using (3) or (4) as the case may be. Then our final estimator for C_i is

$$\hat{C}_i = \frac{1}{N} \sum_{j=1}^{N} \hat{C}_i^{(j)}, \tag{5}$$

where N is the number of simulation runs. In fact, for the policy improvement the interesting quantity is the difference

$$E_{i_1,i_2} = C_{i_2} - C_{i_1},$$

for which we have the obvious estimate

$$\hat{E}_{i_1,i_2} = \hat{C}_{i_2} - \hat{C}_{i_1}. \tag{6}$$

From the samples $\hat{C}_{i_1}^{(j)}$ and $\hat{C}_{i_2}^{(j)}$, $j = 1, \ldots, N$, we can also derive an estimate for the variance $\hat{\sigma}_{i_1,i_2}^2$ of the estimator $\hat{E}_{i_1,i_2}$

$$\hat{\sigma}_{i_1,i_2}^2 = \frac{N \sum_j (\hat{C}_{i_2}^{(j)} - \hat{C}_{i_1}^{(j)})^2 - \left(\sum_j \hat{C}_{i_2}^{(j)} - \hat{C}_{i_1}^{(j)} \right)^2}{N^2(N-1)} = \frac{\hat{S}_{i_1,i_2}^2 - (\hat{E}_{i_1,i_2})^2}{N-1},$$

where $\hat{S}_{i_1,i_2}^2 = \frac{1}{N} \sum_j \left(\hat{C}_{i_2}^{(j)} - \hat{C}_{i_1}^{(j)} \right)^2$.

The choice between the alternative statistics collection methods is based on technical considerations. Though estimator (3) (blocking time) has a lower variance per one simulation run, it requires much more bookkeeping and the variance obtained with a given amount of computational effort may be lower for estimator (4) (blocking events).

4.2 Policy iteration with uncertain state costs

The important parameters of the simulation are the length of the simulation period T and the number of simulation runs N used for the estimation of each C_i. In practice, we are interested in the smallest possible values of T and N in order to minimize the simulation time. However, making T and N too small increases the simulation noise, i.e. error in the estimates $\hat{C}_i$, occasionally leading to decisions that differ from that of the true iteration policy, consequently deteriorating the performance of the resulting algorithm.

No matter how the parameters are selected, some uncertainty in the estimators $\hat{C}_i$ is unavoidable. In order to deal with this uncertainty of the estimators $\hat{C}_i$, we do not blindly accept the action with the smallest estimated cost, but give a special status for the decision which would be chosen by the standard policy. Let us give this action the index 0. Based on the simulations we form estimates $\hat{E}_{0,i}$ for each possible action i. Then, as the decision we choose the action which minimizes the quantity

$$\hat{E}_{0,i} + k \cdot \hat{\sigma}_{0,i}, \tag{7}$$

where k is an adjustable parameter. Note that for $i = 0$ this quantity is equal to 0. Thus, in order for another action i to replace the action 0 of the standard policy, we must have $\hat{E}_{0,i} < -k \cdot \hat{\sigma}_{0,i}$, i.e. we require a minimum level of confidence for the hypothesis $C_i < C_0$. An appropriate value for k has to be determined experimentally.

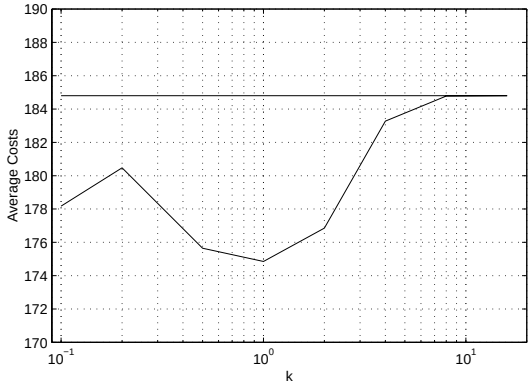

Figure 2: Average cost rate c of the iteration policy as a function of the parameter k. The horizontal line represents the cost rate of the standard policy. The minimum lies in the range $k = 0.5 \ldots 2$ in this case. The set of routes was specifed with $\Delta l = 0$ and $rmax = 4$ explained in 6.1.

If k is too small, wrong decisions are made more frequently. On the other hand too high a value of k prevents the choice of other alternative actions totally. In figure 2 the performance behaviour of a certain system is depicted as a function of k. The horizontal line represents the costs obtained with the standard policy. From the figure it can be seen that once k is higher than about 10 the iteration policy reduces to the standard policy.

4.3 Time complexity of iteration approach

Clearly the simulation of the future, even for a limited period T, at each decision epoch makes this algorithm very time consuming. Assume that a single decision of the standard policy takes a constant time u. Let N be the number of the simulations that are run for each alternative action, A average number of alternative actions per decision (possible RW pairs), λ the total arrival rate to the network (assuming uniform load for simplicity), and T the period covered by one simulation run. Then, the running time of each decision is on the average

$$u_i = A \cdot N \cdot (\lambda T)u = \lambda ANT \cdot u,$$

so the running time is λANT times longer than with the underlying algorithm. Neither λ nor A are parameters of the algorithm. Hence, the tradeoff between the goodness of solution and the running time is defined by choosing the value for product $N \cdot T$.

For example, to get decent results with a simple 11 node network (fig. 3) with moderate load ($\mu = 1$ and $\lambda_k = 0.4$ for all k), about 100 samples (simulation runs) were required, each $1/\mu$ time units long. So the increase in running time was of the order of $10^3 - 10^4$. It is clearly essential that the decisions of the underlying standard policy can be determined quickly.

5 Heuristic Algorithms

Several quick heuristic algorithms for the dynamic RWA problem have been proposed in the literature. Here we briefly present some of them and later study how iteration approach works with them. The first set of algorithms assumes that a fixed set of possible

routes for each connection is given in advance. Some papers refer to this as alternate routing. In practice this set usually consists shortest or nearly shortest path of routes. These algorithms are greedy and accept the first feasible RW pair they find (first-fit).

- *basic* algorithm goes through all the routes in a fixed order and for each route tries all the wavelengths in a fixed order. The routes are sorted in the shortest route first order.
- *porder* algorithm is similar to *basic*-algorithm but it goes through all the wavelengths in a fixed order and for each wavelength tries all the routes in a fixed order.
- *pcolor* algorithm works like *porder* but wavelengths are searched in the order of the current usage instead of a fixed order, so that the most used wavelength is tried first.
- *lpcolor* algorithm is the smartest algorithm. It packs colors, but the primary target is to minimize the number of used links. So the algorithm first tries the most used wavelength with all the shortest routes, then the next often used wavelength and so on. If no wavelength works, the set of routes is expanded to include routes having one link more and wavelengths are tried again in the same order.

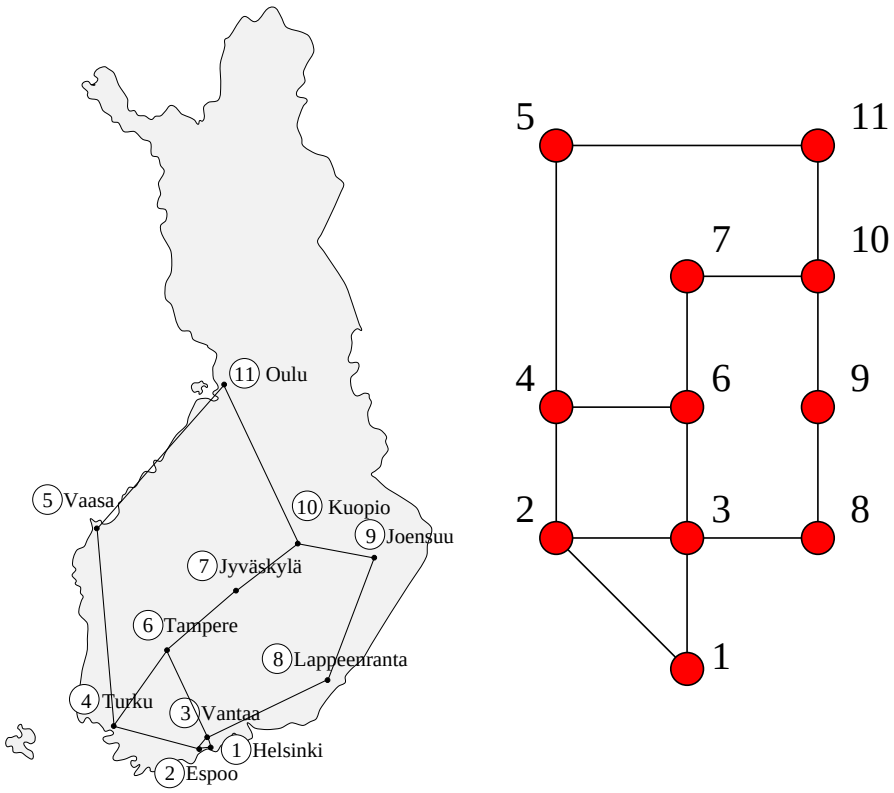

Figure 3: Hypothetical WDM-network residing in Finland.

Another set of heuristic algorithms, adaptive unconstrained routing (AUR) algorithms, are described in [7]. These use dynamic routing instead of fixed set of routes, and are thus a little bit slower.

- *aurpack* is similar to *pcolor*, but without the limitations of a fixed set of routes.
- *aurexhaustive* finds a route with each wavelength (if possible) and chooses the shortest among them, i.e. it is identical to *lpcolor* except that the set of possible routes is not limited.

Set	Description
I	*Uniform case.* The offered traffic between each node pair is uniform with the rate of $\lambda = 0.4$ and the cost of each missed request is equal to 1.0. The average duration of connection is $1/\mu = 1.0$.
II	*Non-uniform costs.* The offered load between each node pair is still uniform, but cost of missed calls differ. A missed connection request to/from node 2 costs 3.0, and connections 1-3, 1-6, 1-7, 6-7 and 6-10 have weight 1.0, and rest of the connections have weight 0.5. So the total cost arrival rate is the same as in the uniform case. The average duration of connections is the same 1.0.
III	*Non-uniform arrival rates.* Similarly here we set the arrival rates of the connections where the other end is node 2 to 1.2, and arrival rates between 1-3, 1-6, 1-7, 6-7 and 6-10 are set to 0.4, while rest of the connection have arrival rate of 0.2. Thus, the total arrival rate of the connection requests to the network stays again the same. The average duration of connections is the same 1.0.

Table 1: The test scenarios: case I is a reference point where everything is uniform and in other two cases either arrival rates or costs are non-uniform.

Thus AUR-algorithms will search for a free route dynamically based on the current state of the network. There is no need to store possible routes (which without any limitations can form a very large set) in advance. This approach is in slight conflict with the iteration approach, where the set of possible actions (RW pairs) is expected to be known in advance.

Also other heuristics are given in [7], e.g. *random* (tries wavelengths in random order) and *spread* (tries least used wavelength first), but they were reported to work worse than the ones described above, and are not further discussed here.

A deficiency of all the presented heuristic algorithms is that they do not take into account the possible additional information about the arrival rates, the distribution of holding times or the priorities of traffic classes (different costs/revenues). Also the duration of the call when it arrives could be known (for example one channel is reserved for certain event which lasts exactly two days). We could of course try to come up with better heuristics which somehow take into account additional information, but that means that we would need a new heuristic policy for each case. On the other hand the first policy iteration automatically adapts to the new situation, because the simulation automatically takes into account all the peculiarities of the system. So, even if the approach is very simple, it is very powerful due to its flexibility.

6 Simulation results

In this section we present some numerical results from simulations. Three test scenarios were created each having different kind of characteristics. Every traffic scenario was based on the small network shown in figure 3. The network was assumed to have 8 wavelengths available on each link and all the links contained one fibre. The test scenarios used in the simulations are listed in table 1. In the first case we have uniform traffic and it is used as a reference point. In the other two cases we have given a special status to the node 2. The special status could arise e.g. in the case where the node represents a gateway to international connections. To facilitate comparison with the uniform traffic case also rates λ_k and expected revenues per call w_k were adjusted so that the offered income rate

,$W = \sum_k \lambda_k w_k$, was kept constant.

It should be recognized that the results for the iteration policy were obtained by two levels of nested simulations. In order to assess the performance of the policy, an outer simulation is run, where connections arrive and leave the network and blocking times or events are recorded. Upon each arrival in this outer simulation, a number of inner simulations are launched from the current state in order to make a comparison between different decision alternatives. Based on this comparison one alternative is chosen and used in the outer simulation, which then continues until upon the next arrival the decision analysis by the inner simulations is again started.

6.1 Selection of routes

The possible routes per node pair (or traffic class) were calculated beforehand. Generally the set of routes is enormous, so some way of pruning it is needed. In this study the set of routes was specified with parameters Δl and *rmax*. Parameter Δl sets the maximum number of extra additional links a route can contain when compared to the shortest route. The second parameter *rmax* defines the maximum number of routes per traffic class, i.e. only the *rmax* first routes are included in set. For example, with Δl=0 and *rmax*=10 only the shortest routes are included, and if there are more than 10 shortest routes for some node pair only the first 10 found are included.

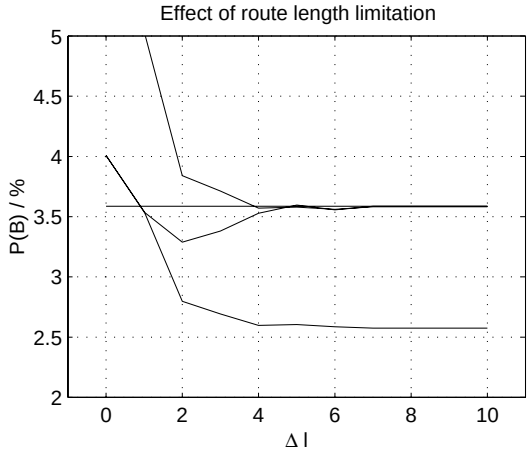

Figure 4: Blocking probability as a function of the parameter Δl. The other routing parameter *rmax* set no limit on the number of routes. At $\Delta l = 3$ the algorithms from worst to best are *basic*, *aurpack*, *pcolor* and *lpcolor*.

Third possible parameter to reduce the running time of the first iteration approach is *maxtest*, which defines the number of alternative actions evaluated against the standard policy. This effectively limits the running time when the load is low in the network and there are plenty of RW pairs available.

In figure 4 the performances of a few heuristic algorithms are presented as a function of the routing parameter Δl. The other routing parameter *rmax* was chosen to be high enough in order not to cause any restriction on the set of routes. The y-axis represents the blocking probability under a uniform load. Clearly too small a set of routes limits the performance but, as can be seen from figure, also too large a set of routes can be a problem for some algorithms. Here the problematic algorithm is *pcolor* which needlessly favors the most used colors at the expense of longer routes, leading to degraded performance.

6.2 Estimation of optimal simulation period

Usually the longer the simulation period T is the better results are obtained. Here we are, however, interested in how current decision affects the results, when the standard policy is used for all later decisions. As was explained before, after a transient period the cost rate $c_t(i)$ is very near to the long time average c of the standard policy. Simulating over a period longer than the duration of the transient thus gives no new information but actually only increases the noise resulting from the stochastic nature of the simulation.

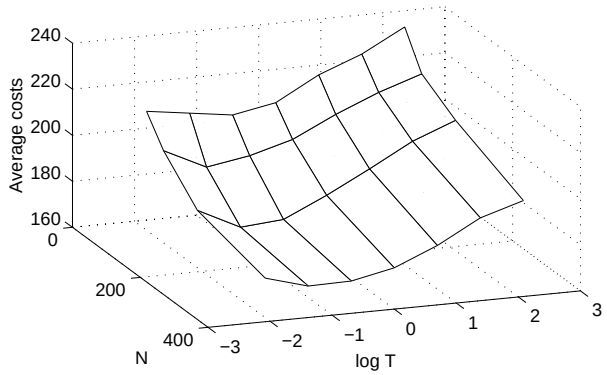

Figure 5: The performance of the algorithm with different simulations periods T and number of simulation runs N. The traffic is uniform and the routing parameters are Δl=0 and *rmax*=4. *basic* is used as the standard policy. Load is $a = 0.4$ for each traffic class. For the reference, the standard policy gives an average cost of 260 for the same arrivals.

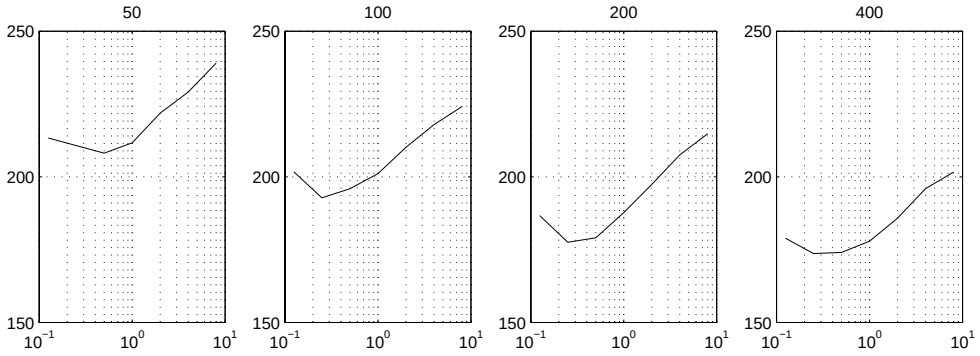

Figure 6: The performance of the algorithm with different simulations periods T with constant number of simulation runs N. The setup is the same as in figure 5 so the graphs represent cuts from the 3D-surface of figure 5

The average costs are depicted in figures 5 and 6 using the first policy iteration with different simulation periods and number of simulations. The offered load to network is $a = 0.4$ for each traffic class. In the mesh figure 5 the x-axis is $\log_{10}$ of the simulation period T and y-axis is the number of simulation runs N. The z-axis represents the average costs.

In figure 6 each subfigure has a fixed number of simulation runs (50, 100, 200 or 400). The x-axis represents the length of simulation period T and y-axis the average costs. As can be seen from figures 5 and 6 the results get worse as the simulation period T grows longer than $0.5 \ldots 1.0$ average holding times. This suggests that the optimal simulation period is about $0.5 \cdot 1/\mu$ in this case.

parameter	value	description
W	22	the total offered cost rate to the network, uniform in cases I and II
λ_{tot}	22	the total offered load to the network, uniform in cases I and III
μ	1.0	the duration of connection, $\sim$Exp(μ)
k	2.0	constant at decision making, defines "certainty"
N	200	the number of simulations run in iteration approach
T_1	$0.25 \cdot 1/\mu$	the length of simulation run in iteration approach, first run
T_2	$0.50 \cdot 1/\mu$	the length of simulation run in iteration approach, second run

Table 2: The running parameters used in test cases.

6.3 Performance of the iteration algorithm under non-uniform traffic

In this section we investigate the performance of the policy iteration starting from different heuristic policies. The parameters used in the simulation are given in table 2. The choice of these parameters represents a tradeoff between the performance and the running time, and was done on the basis of the considerations of the previous sections.

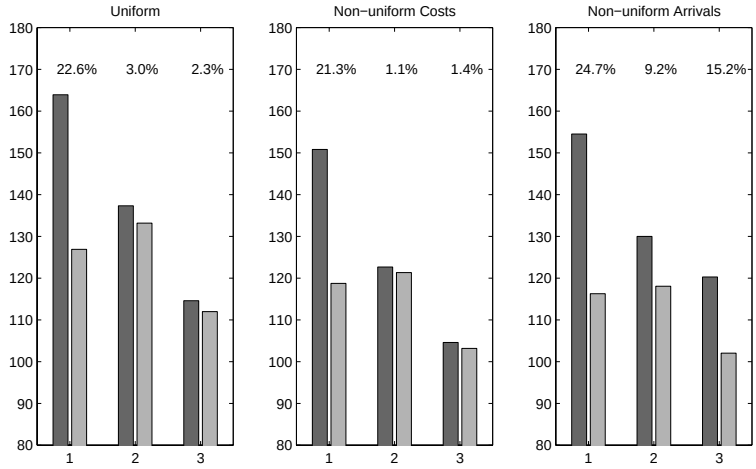

Figure 7: The performance of the algorithm in different test scenarios with *basic*, *pcolor* and *lpcolor* as standard policy. The simulation period is $0.25 \cdot 1/\mu$. Each pair of bars relates to one of these policies, the left bar is obtained with the standard policy and the right bar with the corresponding iteration policy.

As suggested by figure 6 the number of simulation runs N was chosen to be 200 and the simulation period T to be 1/4 or 1/2 times the average holding time. The improvement when the number of samples was increased from 200 to 400 was not significant. Note that the previous simulations were ran with a much smaller set of possible routes per node pair. New routing parameters $\Delta l = 3$ and $rmax = 30$ allow routes that are longer than shortest path in the search space. While the number of routes in these simulations is much larger than in the ones from which the parameters were obtained, it is expected that the same parameters work quite well. As the number of possible actions increased, the value of k was chosen to be 2.0 to be on the safer side (see figure 2).

The first policy approach was studied with these parameters in three different traffic

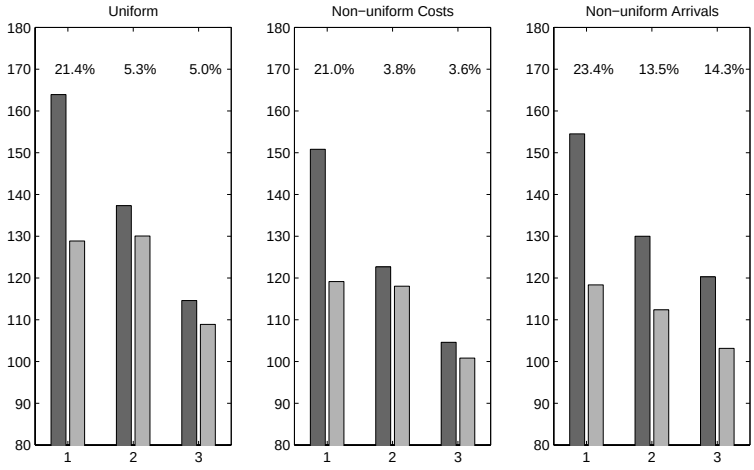

Figure 8: The performance of the algorithm in different test cases with *basic*, *pcolor* and *lpcolor* as standard policy. The simulation period is $0.50 \cdot 1/\mu$.

scenarios presented in the beginning of thus section. The results with different parameters and algorithms are presented in figures 7 and 8. The y-axis represents the average costs obtained from the outer simulation. In figure 7 the simulation period T was 1/4 times the average holding time while in figure 8 the period was extended to 1/2 times the average holding time. The results of standard policies are naturally the same in both figures. Each pane represents one traffic scenario. In the first pane the traffic is uniform, in the second pane the costs are non-uniform and in the last pane the traffic is non-uniform. The bars in figures represent average costs in each case. In each pair of bars the bar on the left represents the result obtained with the standard policy and the bar on the right is the one obtained with the policy iteration. The standard policies from left to right are *basic*, *pcolor* and *lpcolor*. The length of the outer simulation from which the average costs were collected is 200 holding times.

Figures 7 and 8 show that in each case the iteration leads to a better policy, and that the improvement is really notable in the case where *basic* was used as the standard policy. Since *basic* is the worst algorithm, this improvement is not a surprise. Note that in the uniform case the average costs are generally higher than in the other cases. This is probably because a large part of the "important" connections were quite short making the non-uniform case easier to control. It is also worth noting that *basic* policy improved with the policy iteration gave results roughly equal to *pcolor* with the iteration.

In the non-uniform cost case the improvements were generally much less when *pcolor* or *lpcolor* were used as standard policy. Still the difference between *pcolor* and *lpcolor* performance is clear and this suggests that the first policy iteration was not able to come to very close to the optimal policy.

The non-uniform arrival case on the other hand was very favorable for the first iteration approach. The improvement over any heuristic policy was around 10% or higher. This can be explained by the fact that as we are actually sampling the possible future realizations, the important (i.e. more probable) ones are automatically more frequently chosen. So we get a better grasp of the future with fewer samples. In the case where costs differ, similar favoring of more probable realizations does not occur naturally. It would probably be useful to study the applicability of importance sampling method to make more important paths more frequent in the simulations.

7 Conclusions

In this paper we have studied the RWA problem in networks where the offered traffic is non-uniform or different traffic classes have different revenues. The problem was handled in the framework of Markov decisions processes. In particular, we have studied the applicability of the first policy iteration where the relative costs of states are estimated as they are needed by simulations on the fly.

The problem with heuristic algorithms presented in the literature is that they do not take into account the non-uniform traffic or other peculiarities of the system. The first policy iteration is expected to some extent to come over these deficiences, and this was actually the case in both non-uniform test scenarios. The performance improvement obtained by the policy iteration depends on the standard policy one starts with, and in these tests the average cost rate was reduced by about 10% to 20% in most cases. The improvement was not so high in the case where *pcolor* heuristics was used as the standard policy.

The running time of first iteration approach is probably too long for systems where decisions must be made in few seconds. But for slower systems it indeed can be used as an improvement to heuristic algorithms in real time. Even when a long running time makes the approach infeasible in real time, this method can still be used to assess how close the performance of an arbitary heuristic algorithm comes to the optimal policy.

References

[1] B. Mukherjee, *Optical Communication Networks*. McGraw-Hill series on computer communications, McGraw-Hill, 1997.

[2] R. Ramaswami and K. Sivarajan, *Optical Networks, A Practical Perspective*. Morgan Kaufmann Series in Networking, Morgan Kaufmann Publishers, 1998.

[3] A. Willner, "Mining the optical bandwidth for a terabit per second," *IEEE Spectrum*, pp. 32–41, Apr. 1997.

[4] D. Banaree and B. Mukherjee, "A practical approach for routing and wavelength assignment in large wavelength-routed optical networks," *IEEE J. Select. Areas Commun.*, vol. 14, pp. 903–908, June 1996.

[5] S. Baroni, *Routing and wavelength allocation in WDM optical networks*. PhD thesis, University College London, May 1998.

[6] E. Karasan and E. Ayanoglu, "Effects of wavelength routing and selection algorithms on wavelength conversion gain in wdm optical networks," *IEEE/ACM Trans. Networking*, vol. 6, pp. 186–196, Apr. 1998.

[7] A. Mokhtar and M. Azizoğlu, "Adaptive wavelength routing in all-optical networks," *IEEE/ACM Trans. Networking*, vol. 6, pp. 197–206, Apr. 1998.

[8] R. Ramaswami and K. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Trans. Networking*, vol. 3, pp. 489–500, Oct. 1995.

[9] G. Mohan and S. Murthy, "A time optimal wavelength rerouting algorithm for dynamic trafffic in wdm networks," *IEEE J. Lightwave Technol.*, vol. 17, pp. 406–417, Mar. 1999.

[10] H. C. Tijms, *Stochastic Models, An Algorithmic Approach*. John Wiley & Sons Ltd, 1994.

[11] Z. Dziong, *ATM Network resource management*. McGraw-Hill, 1997.

[12] E. Hyytiä and J. Virtamo, "Dynamic routing and wavelength assignment using first policy iteration." to be presented in *ISCC'2000, the Fifth IEEE Symposium on Computers and Communications*, Antibes, Juan les Pins, France, July 2000.