Improving Dynamic Kidney Exchange

Vitor Hadad*

December 2017

Abstract

*Boston College

1 Introduction

Main idea We propose a new dynamic kidney matching algorithm that takes as input a pool of patient-donor pairs and computes, for each cycle that is currently available for matching, a probability that it should be chosen to be cleared today, as well as another probability indicating whether we should not match any more cycles. In order to get these probabilities, our algorithm first produces a prior distribution estimated by observing past choices made by an infeasible optimal algorithm, and then updates these priors with simulations.

Our algorithm faces two main challenges. The first challenge is the input data representation, because the relevant information consists not only of the characteristics of each donor-patient pair, but also the underlying structure of their compatibility graph. For example, we need it to be the case that the same cycle will have different probabilities of being matched depending on what who are all the other patient-pairs in the graph, what characteristics they have, and how compatible their are to each other. Our solution to this obstacle involves using mapping the raw input data and the compatibility graph to points on the euclidean space, and then using these as regressors.

The second challenge is output data representation. As we said above, one of the components of the output is a joint probability distribution over each available cycle, indicating the likelihood that the cycle will be matched today. However, the number of available cycles will be different at each period, so we need an algorithm that is able to output probability distribution over a vector of variable length. In this paper, we experiment with a variety of neural network architectures for this task, and we note that recurrent neural networks provide the best performance.

Main results We test our algorithm in a variety of data-generating processes that we call *environments*. Each environment was inspired by previous work on dynamic kidney matching, and has different rules governing which patient-donor pairs are compatible or what are their relevant observable characteristics. In each environment, we compare the performance of our algorithm, as measured by the number of matched pairs within a long time period, to two benchmarks. The first one is an infeasible optimal algorithm (`OPT`) that knows exactly which patients will be come to the pool and how long their sojourn will be, and is therefore able to compute the maximal matching cardinality. Our second benchmark is `Myopic`, an algorithm that greedily matches and clears the maximal matching at each period, and is used a proxy to the algorithms that are being used in many real life kidney exchanges. Our results show that we were indeed able to improve upon `Myopic`, proving that indeed there might be potential gains to be made from taking dynamic issues into serious consideration.

Our simulations show that there is a sizable performance gap between `Myopic` and `OPT`: for the parameter combinations we have tested, `Myopic` may match only about 75% of the pairs that `Myopic` matches. In addition, this

performance gap widens as we increase the number of maximum cycle length from 2 to 3.

At the same time, we remark that even after our improvements, the gap between `Myopic` and `OPT` resists. Therefore, in this paper we also invite other researchers to consider alternative algorithms for dynamic kidney exchange, or produce theoretical upper bounds for them. We hope that the new methods proposed in this paper serve as further inspiration for the field.

2 Background and literature review

Since its proposal in the medical literature over three decades ago by **?**, kidney exchange has been the subject of a great deal of attention in the academic literature. Throughout the past twenty years, developments in microeconomic theory and operations research and have informed, reshaped and improved how living donor transplants are conducted in many countries. More recently, computer scientists have also started to focus on the production of better and faster algorithms for kidney exchange.

Since this paper sits in the intersection between matching theory and computational economics, and uses methods drawn from computer science and machine learning. Let us take a brief look at these fields in turn.

2.1 Kidney Exchange

Definitions ¿HLA compatibility can also be defined in terms of mismatch acceptabilitymismatches in allosensitized transplant candidates that result in a negative crossmatch [74]. In this context, unacceptable mismatches are antigens reacting with antibody detectable in the patient sera, whereas acceptable mismatches are those with no detectable antibody

Early years In economics, the literature on kidney exchange is started by **?**, who provided a kidney exchange mechanism inspired by the *housing* problem studied in the literature of mechanism design as in **?** and later **?**. While their mechanism had significant virtues as being strategy-proof and Pareto-efficient, it also relied on large number of exchanges being conducted simultaneously, and drew criticism for making assumptions about heterogeneous preferences over kidneys. Responding to these assessments, a subsequent work by **?** focused on logistically practical mechanisms that used only 2-way exchanges and also had patients indifferent over all compatible kidneys. In a later paper, **?** demonstrated via simulations that allowing for 3-way exchanges in addition to 2-way exchanges could increase the cardinality of matched pairs by a great deal, but also showed that larger cycles would only bring about modest improvements.

Recent advances in static matching problems In the second half of the oughties, both in academic literature and medical practice the focus began to shift from exchanges via cycles to exchanges emanating from *non-directed donors* (NDD) – altruistic donors who have no patient attached to them and are willing to donate their organs to anyone who would need it. Such exchanges yield a *chain* of transplants that begins with the NDD and may either terminate with the last pair donating to a waitlist recipient in a kidney registry, or not terminate at all and have the last pair's donor await an opportunity to become a future living donor. The former kind was dubbed *domino paired donation* by **?**, and all its transplants occur simultaneously. In the latter kind, transplants occur may staggeredly over several months, hence it was named *non-simultaneous, extended, altruistic donor chain* by **?**.[1]

A significant amount of attention was devoted in the next years to these NEAD chains, including notably **?** and **?**, whose results show that NEAD chains benefit highly sensitized patients in sparse pools of moderate size, because they decrease the need for simultaneous double-coincidence of wants in the exchange market. In fact, in recent years **?** reported that "NEAD chains are responsible for the majority of successful kidney transplants conducted via kidney exchange at both the [Allied for Paired Donation] and within other major exchange programs".

Dynamic kidney exchange Meanwhile, a small subliterature started focusing on the problem of *dynamic kidney exchange* that concerns this paper. In a seminal paper, **?** posited a continuous-time model where pairs arrive according to a Poisson arrival rate, enter the pool whenever their donor is incompatible with them, and from thereon suffer a constant cost of waiting nver was able to derive dynamic mechanism that depends produces optimal n-way exchanges by leveraging additional simplifying assumptions, including that pairs do not leave the pool unless they are matched, and that pairs within the pool are only blood-type incompatible. These assumptions render all pairs homogeneous, which dramatically decreases the dimension of the state space and allows for an easily interpretable solution.

? note that the graphs of real-life pools tend to be sparse and filled with pairs that are either easy or extremely hard to match. In order to model this particular feature, they postulate a sparse heterogeneous random graph model containing only two such types, and propose a greedy algorithm that waiting until there are a certain number of each type, and then matches as many as pairs as possible. In a different vein, **?** introduces a simple model of stochastically arrival and departure. In their setting, they show that greedy algorithms that cleverly explout the time to match can can perform close to infeasible optimal benchmarks, even if these algorithms are ignorant about the global structure of the graph.

[1] On occasion, NEAD chains have also been called *Never-ending altruistic donor chains*, as relayed by (**?**, p. 235-6).

In the computer science and operations research literature, **?** and more recently **?**, **?** and **?** have focused on producing scalable combinatorial programming algorithms that can take on tatic matching problems with large graphs and allowing for both chains and cycles of moderately large length. However, a series of papers starting with **?** and followed by **?** and **?** have dealt with the dynamic kidney problem by *weighted myopia*. Their idea is to prevent wasteful matchings (e.g., an O-donor to an AB-patient) by artificially introducing negative weights to graph components containing them. In **?**, these optimal weights are computed from simulations involving historical data.

Miscellaneous Kidney exchange is a rich problem that can be studied from several different angles. For example, it is a special case of barter exchange, [[[Barter]]]. [[[Hospital incentives]]] [[[Medical literature]]]

2.2 Machine learning and Artificial Intelligence

2.2.1 Neural Networks

Neural networks are created by composing simpler parametric building blocks such as linear functions and simple nonlinear ones**?**. Although they have existed for more than a half-century, in recent years they have received an enormous amount of attention in the statistical, computer science and optimization literatures. In this work, we will use *multilayer feedforward networks*, *recurrent neural networks* (with and without an extra property named *attention*) and *graph convolutional networks*. While a comprehensive survey on the motivation and use-cases of all these types of neural networks is outside the scope of this work, in the next paragraphs we will give an abridged overview of their definition and their immediate application to our problem.

Feedforward Neural Networks Also called *multilayer perceptrons*, these functions process each input

Recurrent Neural Networks and Sequence-to-sequence models

Embedding non-euclidean domains

2.2.2 Search algorithms

3 Models, Environments and Data

A **dynamic kidney exchange problem** is a discrete-time Markov Decision Process (MDP) described by the tuple (S, A, P, R), where

- $S = (V, A, X)$ is a directed graph representing the current *state*: the set V is the finite set of patient-donor pairs and their observable characteristics; and an adjacency matrix A whose entry (i, j) is positive whenever the donor of vertex i can donate to the patient of vertex j. The set X is a set of

- Act is the finite set of available *actions* when the current state is S. In our applications, this is the set of 2- or 3-cycles available to be cleared.

- P is the *transition probability* over the next states given current state and taken action.

- R is the *reward function* indicating how desirable it is to take an action $a \in Act$ when the state is S.

The space where a dynamic kidney exchange problem tuple lives is called a **kidney exchange environment**. Informally, an environment is the set of rules or configurations that govern, for example, whether we can match 2-cycles or 3-cycles per period, or which characteristic are relevant and available for the decision agent to observe. Before we describe each environment in detail, here are some common assumptions.

Poisson entry, Geometric death At time point, the number of new incoming pairs is drawn from the $Poisson(r)$ distribution, where $r \in \mathbb{N}$ denotes the *entry rate*, and equals the expected number of entrants per period. Upon entrance, each pair independently draws the length of their sojourn from the $Geometric(d)$ distribution. The parameter $d \in \mathbb{R}$ is the *death rate*, and its reciprocal $\frac{1}{d}$ is the expected sojourn length. We note that, due to the memoryless property[2] of the Geometric distribution, the amount of time a pair has waited in the pool gives us no information about how much time they have until their death.

Observables In every environment, at least two characteristics are relevant and observable: the major blood type group (A, B, O or AB); and how long the pair has been in the pool.

0-1 Preferences In reality, some exchanges are more or less desirable than others, either for ethical concerns or because of predicted health benefit. For example, it is common for pediatric patients and previous organ donors receive higher priority, as do exchanges involving patients with no HLA mismatch. However, in this work we abstract from these concerns and consider every exchange to be equally desirable, so long as it is available. This is the same assumption as used in **?**.
Now, let's see each of the different environments in turn.

[2]If $X \sim Geometric(p)$, then $P(X > t + s|X > s) = P(X > t)$

3.1 ABO Environment

The *ABO environment* is the barest: each pair in the pool is solely characterized by their entry time, the length of their sojourn, and the ABO blood types of its patient and donor. Compatibility between two pairs is also decided only on blood-type compatibility. However, we make an assumption previously used in **?** and allow for incompatibility between a donor and their own patient. The reason for this additional assumption is that, if there were truly no tissue type compatibilities, we would never see pairs of type (AB,·), (·, O), or (A,A), (O,O), (B,B), and (AB,AB), since their donors would be automatically compatible with their patients and they would never participate in an exchange. Let's explain the effect of this assumption in more detail.

The distribution of blood types in the US population is roughly O:48%, A:36%, B:11%, AB:4%. If we initially independently draw two people from this distribution and they form, say, an (A,B) pair (which happens with probability $0.36 \cdot 0.12 \approx 0.0432$), then we allow them to the pool immediately, since their B-donor is blood-type incompatible with their A-patient. However, if they happen to form an (A,O) pair (with probability $0.36 \cdot 0.48 \approx 0.172$), then they should only enter the pool if the donor and patient are tissue-type incompatible, otherwise the O donor would immediately donate their kidney to the patient and they would not need to enter the exchange pool at all. So we assume the probability of a positive crossmatch is $p_c = 0.11$ as in **?**, then the probability that two people form an (A,O) and enter the pool becomes in fact $0.36 \cdot 0.48 \cdot 0.11 \approx 0.019$. By computing the probabilities of each pair in this manner and then normalizing so that they add up to one, we arrive at the numbers displayed on Table 1.

Pair blood type	Probability
(O, O)	0.058689
(O, A)	0.373803
(O, B)	0.158257
(O, AB)	0.042669
(A, O)	0.041119
(A, A)	0.028809
(A, B)	0.110888
(A, AB)	0.029899
(B, O)	0.017410
(B, A)	0.110888
(B, B)	0.005160
(B, AB)	0.012660
(AB, O)	0.004690
(AB, A)	0.003290
(AB, B)	0.001390
(AB, AB)	0.000380

Table 1: Blood type probabilities in the ABO environment

3.2 RSU Environment

The *RSU* environment was inspired largely by the simulation setup in **?**. In these works, in addition to the blood type, a pair is also endowed with a panel reactive antibody (PRA) level that governs the probability of a crossmatch with a random donor. The lower the PRA, the higher the number of potentially compatible pairs.

The simulation process is as follows. Initially, we draw a pair in the same manner as in the ABO environment. Next, we draw if the patient is a female (with probability around 41%), and if so we also draw whether her donor is her husband (spouses comprise about 49% of donors). Finally, we draw a PRA level for the patient (Low: 70.1%, Medium: 20%, High: 9.9%). This PRA level determines the probability that they can receive a kidney from any donor, including their own: patients with low PRA have a 5% probability of positive crossmatch with a random donor; patients with medium PRA have a 45% chance, and patients with high PRA have a 90% chance of a crossmatch. If a patient is bloody or tissue-type incompatible with their own donor, they enter the pool. In addition, if the patient is female and her husband is the donor, the probability of positive crossmatch for low, medium and high PRA patients goes up to 28.75%, 58.75% and 92.25%. This last adjustment reflects the fact that women tend to produce antibodies against their husbands' antigens during pregnancy.

Once in the pool, the pair immediately forms directed edges with the existing pairs, again following the patient PRA distribution. The resulting random

textwidthtextwidth../Users/vitorhadad/Documents/kidney/matchir

Figure 1: Computed patient and donor cPRA.

graph is akin to a Erdös-Rényi $G(n, p)$ random graph, except that the edge-forming probability is heterogeneous across different pair types.

3.3 OPTN Environment

Here we use historical data collected by the United Network for Organ Sharing (UNOS) data provided in the Standard Research and Analysis (STAR) dataset. The STAR dataset contains information from all patients that were ever registered to the kidney waiting list in the United States for the past three decades, as well as from all living donors that actually went to transplant. We excluded individuals with missing HLA profile information, we had access to a detailed historical dataset containing 117813 patients and 9337 donors.

The data set about was used in two ways. First, we created an artificial data set by randomly drawing patients from the historical waiting list, drawing donors from the historical living donor list, and checking for blood- and tissue-type compatibility as explained below. Compatible pairs were discarded. We iterated in this manner to construct a dataset of about one million incompatible pairs.

Second, we used the historical data set to calculate two additional variables, a *patient cPRA* and a *donor cPRA*.[3] As explained before, the patient cumulative panel reactive antibody, or cPRA, represents a measure of patient tissue-type incompatibility with a random donor, the donor cPRA is a novel measure of the opposite direction – how frequently a donor is tissue-type incompatible with a random patient. We computed both measures empirically in our data set, using the definiton of compatibility below.

Compatibility A donor and a patient are deemed *compatible* if they are blood-type compatible and tissue-type compatible. The latter is true when the donor's HLA profile exhibits no antigens that are unacceptable for the patient in any of the A, B, Bw, C, DR, DPB, DQ, and DQA loci.

Simulation During a simulation, pairs are drawn from this artificial data set. Once a pair enters the pool, it immediately forms directed edges with existing compatible pairs.

Observable characteristics For each pair, the agent observes the entry time, the waiting time, and 375 dummies corresponding to blood type and HLA profile of patient and donor.

[3]We thank Itai Ashlagi for the suggestion of a donor cPRA.

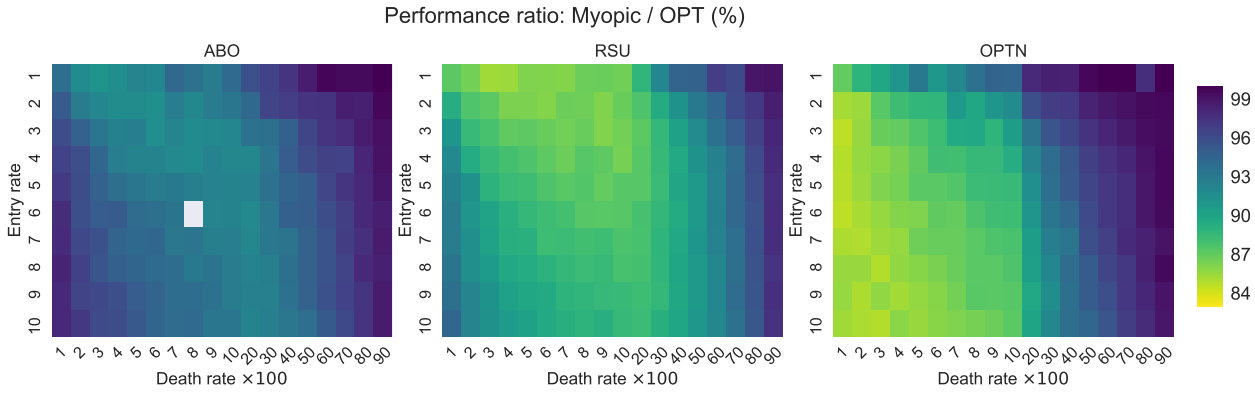

Figure 2: `Myopic` matches roughly the same number of pairs per period as `OPT` when the death rate is high (moving rightwards on the graphs), or when entry rate is high (moving downwards on the graphs).

3.4 Benchmarks

In this work, we focus on maximizing the average number of matched pairs over T periods, where T is a large number relative to the rates of patient entry and death.

3.4.1 Myopic

The *myopic* algorithm finds the maximal number of matchings at every period, and clears it immediately. It disregards all observable characteristics of each pair, and in particular it disregards that some pairs might be useful to keep certain pairs may be easier or harder to match. In doing so, it may forgo the opportunity of matching a hard-to-match patient today, or postpone an easy-to-match pair for later.

3.4.2 Infeasible Optimal (OPT)

The infeasible optimal algorithm (henceforth OPT) does away with the uncertainty arising from the temporal structure of the problem: it knows exactly which pairs will come into the pool, as well as their arrival, and the duration of their sojourns. This algorithm is essentially a large static kidney matching problem, except that the set of available exchanges has the additional constraint that a cycle cannot exist between two vertices if their sojourns fail to overlap.

3.4.3 Empirical comparison between myopic and OPT

Figure **??**

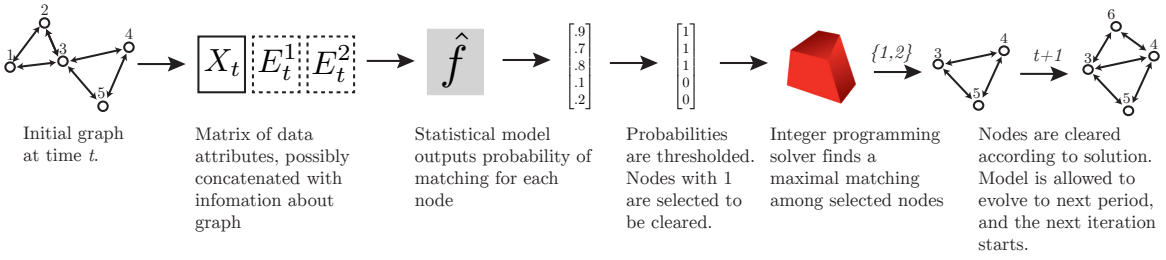

Figure 3: Direct prediction methods.

3.5 Algorithms

We divide our algorithms into three classes: *direct prediction* methods, *multi-armed bandit* methods and *reinforcement learning* methods. They are explained next.

3.5.1 Direct prediction

Dynamic and static algorithms can split their work: the former can determine which pairs should be matched at each period, while the latter decides how they should be matched among themselves. The *direct prediction* exploits this insight, essentially reducing the problem to a classification task: at each period, they aim to produce a 0-1 label for each node indicating whether is should be matched in this period (1) or left for later (0). Selected nodes are then passed to a static solver that finds the maximal matching among them. Once these nodes are cleared, time evolves to the next period. This is illustrated in Figure 3.

In order to produce data to feed into our "classifier", each environment was simulated and solved using OPT, with each pair's observable characteristics as regressors, and a binary variable indicating whether they were matched or not. Each resulting artifical dataset of approximately 2 million observations was then fed to a series of predictive algorithms, including penalized logistic regression**?**, support vector machines with radial basis function kernels**?**, random forest classifiers**?**, gradient tree boosting classifiers**?** and deep feedforward networks**?**. The performance of these algorithms as classification methods is compared in Table **??**.

[[[The poor performance?? could be attributable to not having represented the data properly etc. Non-euclidean spaces big challenge etc]]]

In order to represent information about the graph and how the nodes fit within it, we augment the original dataset using one or both of the following. First, with vertex properties for each node such as the number of in and out-edges, average neighbor degree, some centrality measures (betweenness, in-degree, out-degree, harmonic, closeness) and more exotic indices such as the core number and pagerank of each node**?**. Second, we augment the data using the *node2vec* graph embedding algorithm of **?**, a continuous feature represen-

tation algorithm inspired by the skip-gram model.[[[More on this?]]]. These algorithms will produce auxiliary matrices E_t^1 and E_t^2 that are concatenated to the original data regressor matrix X_t, and the entire augmented data set can again be fed to the predictive algorithms above. Their performance is compared on the bottom half of Table **??**.

3.6 Multi-armed bandits

We have access to the data-generating process (*environments*) and an algorithm that is able to find the best matching for any data in hindsight (`OPT`). Therefore, in principle we could estimate the best choice of cycle to clear by simulating several periods ahead in the future, running `OPT`, and measuring the performance of our choice under any desired criterion.

The approach outlined above turns out to be naive and incomplete, but it essentially contains the insight under which we will be working in this section. It is naive because simulations are computationally expensive, and in practice we cannot repeat them enough times get reliable estimates of the average performance for each cycle choice, especially in large graphs. It is also incomplete because it does not specify exactly what is the best information we should extract from `OPT` results. In order to solve the first problem, we leverage theory and algorithms from the multi-armed bandit (MAB) literature. For the second, we propose a secondary objective based on what we call *pseudo-rewards*.

Our procedure is illustrated in Figures 4 and 5. At the beginning of the period t, the agent receives a set of cycles C that are available to be cleared. If C is empty, nothing happens and we move to the next period. Otherwise, the agent then picks a cycle $c \in C$ and simulates the future, including new entries and deaths, up to a horizon h. Next, `OPT` is run twice, once normally, and once with the additional constraint that c be removed today. The size of the resulting matching in these two scenarios is compared. Naturally, the constrained version of `OPT` cannot achieve anything better than its unconstrained counterpart, but it might get to be equal. If it is, the agent receives a *pseudo-reward* of one, otherwise it receives zero. This process is repeated: at each iteration ℓ, a cycle c_ℓ is chosen and its pseudo-reward $r_{c,\ell}$ is revealed. When a preset computational budget of $L(|C|)$ iterations is hit, the agent then analyses the whole history of cycle choices and pseudo-rewards $H_t = \{(c_\ell, r_{c\ell})\}_{\ell=1}^{L}$, and decides whether to match one of the cycles or move on to the next period. If a cycle c is chosen it is immediately cleared, however the environment does not evolve to the next period yet. Instead, the history H_t is discarded the procedure is repeated again with a reduced set of actions $C' \subset C$ that produces a new history H'_t and so on, until either there are no more available choices or the agent decides to allow the environment to move on to time $t + 1$. When that at last happens, entries and deaths are revealed, the agent receives a new set of cycles, and the process begin anew. All past information is ignored.

Two important details were left out of the explanation above. First, how

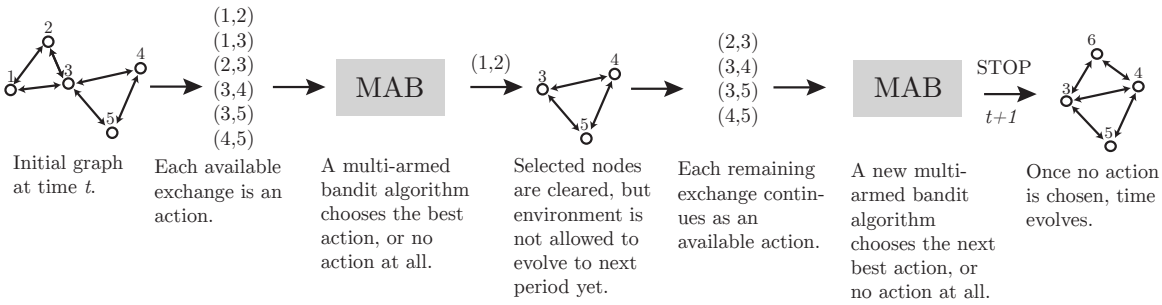

Figure 4: Multi-armed bandit methods.

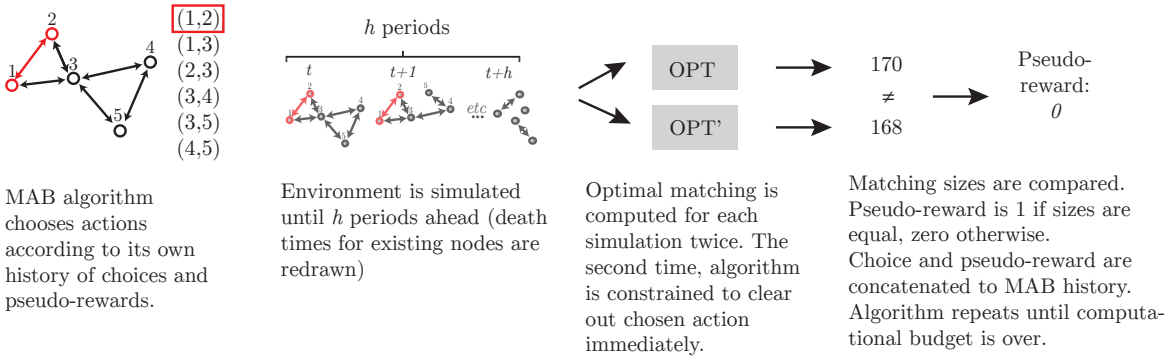

Figure 5: Pseudo-reward computation in one step of a multi-armed bandit algorithm.

does the agent chooses the next cycle to test? Second, how does it decide which cycle to choose (or no cycle at all)?

Let c^* be a cycle that maximizes expected pseudo-rewards during one round of the algorithm:

$$c^* \in \arg\max_c E[r_{c\cdot}]$$

A *multi-armed bandit* algorithm is a procedure to minimize *regret*

4 Results

4.1 Short-run gains

5 Discussions and Extensions

HLA mistmatch – doctors are willing

6 Conclusion

7 Acknowledgements

This work was supported in part by Health Resources and Services Administration contract 234-2005-37011C. The content is the responsibility of the authors alone and does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.

Tables

Figures

Appendix A. Placeholder