Constrained Randomized Shortest Path Problems

(draft manuscript submitted for publication and subject to changes)

Bertrand Lebichot[1], Guillaume Guex[1], Ilkka Kivimäki[1,2] & Marco Saerens[1*]
[1]ICTEAM and Machine Learning Group (MLG)
Université catholique de Louvain (UCL)
[2]Department of Computer Science
Aalto University, Helsinki, Finland

December 3, 2017

Abstract

This work extends the textbfrandomized shortest-path framework (RSP) introduced in [**?**], interpolating between shortest-path and commute-cost distances, in two directions. First, it shows how to deal with equality constraints on transition probabilities and derives a generic algorithm for solving the problem using Lagrange duality. Second, it derives a simple algorithm to compute the optimal, mixed, policy solving simple randomized Markov decision problem by considering the system as a constrained RSP on a bipartite state-actions graph G_{b}. xxxx The constrained RSP considers the system of paths connecting the initial state to the final, goal, state. In this context, a (randomized, mixed) **policy** corresponds to the assignment of a probability distribution on this set, providing the probability of choosing any path to the goal state. The **optimal** policy is obtained by minimizing the free energy (expected cost plus relative entropy) of the system with respect to these probabilities, and is a standard Gibbs-Boltzmann distribution with virtual costs introduced on the constrained edges in order to satisfy the constraints. The mixed, randomized, policy is then easily obtained from these probabilities by exploiting Lagrange duality. The resulting algorithm allows to balance exploitation and exploration in an optimal way by interpolating between a pure random behavior and the optimal, deterministic, policy. It is also shown that the mixed policy can straightforwardly be obtained by iterating the Bellman's value iteration equations in which the minimum operator is replaced by a soft minimum, as in unconstrained RSP [**?**, **?**]. Interestingly, the derived "soft" value iteration algorithm is closely related to dynamic policy programming [**?**, **?**] as well as "Kullback-Leibler" and "path integral" control [**?**, **?**, **?**, **?**, **?**, **?**] and similar to the exploration strategy recently introduced in [**?**, **?**]. Simulation results on an illustrative example shows that the model behaves as expected.

[*]Marco Saerens is also a Research Fellow of the IRIDIA Laboratory, Université Libre de Bruxelles.

1 Introduction

1.1 General introduction

The present work aims to study randomized shortest-path problems with *equality constraints* on the transition probabilities from a subset of edges, in the context of a single source and a single destination. This allows to fix some transition probabilities and then finding the optimal policy compatible with these probabilities.

It therefore extends previous work dedicated to randomized shortest paths (RSP, [**?**, **?**, **?**]), and initially inspired by stochastic traffic assignment studied in transportation science [**?**]. The model can be described informally as follows. Suppose we have to find the optimal policy, minimizing expected cost for reaching a destination node from a source node in a network, where costs are associated to local decisions/actions. Usually, deterministic and stochastic shortest-path algorithms provide pure deterministic policies: when standing in a given state k, we just choose the best action u leading to the minimal expected cost.

In the present work, we investigate the possibility of optimally **randomizing** the policy while fixing a subset of transition probabilities: the agents can choose several actions, according to some probability distribution. Randomizing the policy thus introduces a continual exploration of the network. As in the standard RSP framework, the degree of randomness can be controlled by a **temperature parameter** allowing to interpolate between the least-cost solution given by the optimal shortest-path algorithm and a random behavior provided by a predefined, reference, random policy.

In practice, randomization corresponds to the association of a probability distribution on the set of admissible actions in each node ([**?**], choice randomization or mixed strategy). If no randomization is present, only the best policy is exploited. Randomization appears when this probability distribution is no more peaked on the best choice: the agent is willing to sacrifice efficiency for exploration. Note that randomized choices are common in a variety of fields [**?**]; for instance game theory (mixed strategies; see for instance [**?**]), computer sciences [**?**], Markov games [**?**], decision sciences [**?**], etc.

A comprehensive related work and a detailed discussion of the reasons for randomizing the policy can be found in [**?**], and is summarized here:

- If the environment is changing over time (non-stationary), the system could benefit from randomization by performing continual exploration.
- It is sometimes of interest to explore the environment, such as in reinforcement learning.
- A deterministic policy would lead to a totally predictable behavior; on the contrary, randomness introduces unpredictability and therefore renders interception more difficult. Randomization (mixed strategies) has proven useful for exactly this reason in game theory.
- A randomized policy spreads the traffic over multiple paths, therefore reducing the danger of congestion.

- ► In some applications, like in social networks analysis, computing a distance accounting for all paths – and thus integrating the concept of high connectivity – could provide better results than relying on the optimal, shortest, paths only.
- ► In computer gaming, it is often desirable to be able to adapt the strength of the digital opponent, which can easily be done with the introduced model.

In this context, the randomness associated to paths connecting the initial node and the goal node is quantified by the relative entropy, or Kullback-Leibler divergence (see, e.g., [**?**]), between the probability distribution on the paths and their likelihood according to a reference random walk on the graph, usually a uniform distribution on the set of available actions. This relative entropy captures the degree of randomness of the system.

The optimal, randomized, policy (assigning a probability distribution on the set of actions available on each state) is then obtained by minimizing the free energy (the expected cost plus the Kullback-Leibler divergence), subject to some equality constraints on the transition probabilities provided by the environment – the probabilities of jumping to a given state after having chosen an action – which have to be verified exactly. Based on this formalism, a first, generic, algorithm for solving constrained randomized shortest paths problems is developed by exploiting Lagrange duality.

Then, as an application example, this framework is used in order to solve Markov decision problems for providing mixed, randomized, policies. **Markov decision problems** [**?**, **?**, **?**, **?**, **?**], also called **stochastic shortest-path problems** [**?**, **?**], are currently used in a wide range of application areas including transportation networks, medical imaging, wide-area network routing, artificial intelligence, to name a few (see, e.g., [**?**, **?**, **?**, **?**, **?**]).

A simple, easy-to-implement, **soft value iteration** algorithm solving the problem is derived and its convergence to a fixed point is proved. Interestingly, this "soft" value iteration algorithm is closely related to dynamic policy programming [**?**, **?**] as well as Kullback-Leibler and path integral control [**?**, **?**, **?**, **?**, **?**, **?**], and similar to the exploration strategy recently introduced in [**?**, **?**].

This shows that the recently proposed exploration strategy in [**?**, **?**], developed in parallel to this work, is globally optimal in our setting in the following sense: it minimizes expected cost subject to constant relative entropy of paths probabilities, when the goal state is absorbing and reachable from any other state. Interestingly, as in [**?**, **?**], the softmax value iteration algorithm extends the Bellman-Ford value iteration algorithm by simply replacing the minimum operator by a soft minimum operator.

Note that still another way of solving the problem was developed in [**?**], but this algorithm is not included here because it is less generic. For a comprehensive related work about randomized policies, see [**?**, **?**].

1.2 Contributions and organization of the paper

In brief, this work has four contribution:

- ▶ It extends randomized shortest-paths to problems with constrained transition probabilities on a subset of nodes.
- ▶ The constrained randomized shortest-paths framework is applied to standard Markov decision problems in order to provide optimal randomized policies.
- ▶ A simple, easy-to-implement, soft value iteration algorithm is derived for obtaining optimal randomized policies.
- ▶ Simulations on concrete problems show that the algorithms behave as expected.

What concerns the organization of the paper, Section 2 introduces the randomized shortest paths framework. Section 3 considers randomized shortest path problems with constraints on transition probabilities. In Section 4, the standard Markov decision problem is recast as a constrained randomized shortest path problem on a bipartite graph and an algorithm is proposed for solving it. Moreover, a simple, more efficient, soft value iteration algorithm is also developed. Section 5 shows some simulation examples and Section 6 is the conclusion.

2 The standard randomized shortest path framework

Our formulation of the problem is based on the **randomized shortest path** (RSP) framework defining a dissimilarity measure interpolating between the shortest path distance and the commute-time distance in a graph [?, ?, ?]. This formalism is based on full paths instead of standard "local" flows [?].

We start by providing the necessary background and notation. Then, we proceed with a short summary of the randomized shortest path formalism, before introducing, in the next section, randomized shortest paths with constraints on the transition probabilities.

2.1 Some background and notation

We consider a weighted directed graph or network, G, with a set of n nodes $\mathcal{V}$ (or vertices) and a set of arcs $\mathcal{E}$ (or edges). The graph is represented by its $n \times n$ adjacency matrix $\mathbf{A}$ containing binary values if the graph is unweighted or non-negative affinities between nodes in the case of a weighted graph. To each edge linking node i to node j, we also associate a non-negative number c_{ij} representing the immediate cost of following this edge. The costs should be non-negative and are gathered in matrix $\mathbf{C}$.

Moreover, a **reference random walk** on G is defined in the usual manner. The choice to follow an edge from node i will be made according to a probability distribution (transition probabilities) defined on the set $\mathcal{S}ucc(i)$ of successor nodes of i. These transition probabilities, defined on each node i, will be denoted as

$$p_{ij}^{\text{ref}} = p^{\text{ref}}(s(t+1) = j | s(t) = i) = \frac{a_{ij}}{\sum_{k \in \mathcal{S}ucc(i)} a_{ik}} \quad (1)$$

where a_{ij} is element i, j of the adjacency matrix and $s(t)$ is a random variable containing the node visited by the random walker at time t. Furthermore, $\mathbf{P}_{\text{ref}}$ will be the matrix containing the transition probabilities p_{ij}^{ref} as elements. For consistency, if there is no edge between i and j, we consider that c_{ij} takes a large value, denoted by ∞; in this case, the corresponding transition probability is equal to zero, $p_{ij} = 0$.

Finally, in this work, we will consider that there is a unique absorbing, killing, node which will be the last node n – the goal node. Any transition from this node is forbidden, that is, $p_{nj}^{\text{ref}} = 0$ for all j.

2.2 The standard randomized shortest path formalism

The main idea is the following. We consider the set of all **hitting** paths, or walks, $\wp \in \mathcal{P}$ from node 1 to the (unique) absorbing, or hitting, node n on G. We further assume than this absorbing node n can be reached in a finite number of steps from each other node in the graph. Each path $\wp$ consists in a sequence of connected nodes starting in node 1 and ending in n. Then, we assign a probability distribution $\mathrm{P}(\cdot)$ on the set of paths $\mathcal{P}$ by minimizing the generalized **free energy**[1] of statistical physics [?, ?, ?],

$$\left| \begin{array}{ll} \underset{\{\mathrm{P}(\wp)\}_{\wp\in\mathcal{P}}}{\text{minimize}} & \phi(\mathrm{P}) = \displaystyle\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\tilde{c}(\wp) + T \sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) \log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) \\ \text{subject to} & \sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) = 1 \end{array} \right. \tag{2}$$

where $\tilde{c}(\wp) = \sum_{\tau=1}^{t} c_{s(\tau-1)s(\tau)}$ is the total cumulated cost along path $\wp$ when visiting the sequence of nodes, or states, $(s(\tau))_{\tau=0}^{t}$ and t is the length of path $\wp$. Furthermore, $\tilde{\pi}(\wp) = \prod_{\tau=1}^{t} p_{s(\tau-1)s(\tau)}^{\text{ref}}$ is the product of the transition probabilities (see Equation (1)) along path $\wp$ – the likelihood of path $\wp$. It defines a **reference** probability distribution over paths as $\sum_{\wp\in\mathcal{P}} \tilde{\pi}(\wp) = 1$ [?, ?].

The objective function in Equation (2) is a mixture of two dissimilarity terms with the temperature T balancing the trade-off between them. The first term is the expected cost for reaching destination node from source node (favoring shorter paths – *exploitation*). The second term corresponds to the relative entropy, or Kullback-Leibler divergence, between the path probability distribution and the path likelihood distribution (introducing randomness – *exploration*). When the temperature T is low, shorter paths are favored while when T is large, paths are chosen according to their likelihood in the random walk on the graph G. Note that we should normally add non-negativity constraints on the path probabilities, but this is not necessary as the resulting quantities will automatically be non-negative (the relative entropy forbids negative values). Note that, instead of minimizing free energy, it is equivalent to minimize expected cost subject to a fixed relative entropy constraint [?, ?, ?].

This argument, akin to maximum entropy [?, ?], leads to a **Gibbs-Boltzmann**

[1] Alternatively, we can adopt a maximum entropy point of view. Moreover, the free energy could also be defined as $\phi(\mathrm{P}) = \sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)(\tilde{c}(\wp) - c^*) + T \sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) \log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)$ where c^* is the shortest path cost from starting node 1 to destination node n. In this case, costs are computed relatively to the shortest path cost. This choice leads to exactly the same probability distribution over paths (Equation (3)).

distribution on the set of paths (see, e.g., [**?**, **?**] for a detailed derivation),

$$\mathrm{P}^*(\wp) = \frac{\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]}{\sum\limits_{\wp'\in\mathcal{P}} \tilde{\pi}(\wp')\exp[-\theta\tilde{c}(\wp')]} = \frac{\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]}{\mathcal{Z}} \tag{3}$$

where $\theta = 1/T$ is the inverse temperature and the denominator $\mathcal{Z} = \sum_{\wp\in\mathcal{P}} \tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]$ is the **partition function** of the system. This defines the **optimal mixed policy** in terms of *probabilities of choosing a path*, $\mathrm{P}^*(\wp)$. It can be shown that this set of path probabilities is exactly equivalent to the ones generated by a Markov chain with biased transition probabilities favoring shorter paths, depending on the temperature T [**?**].

2.3 The minimum free energy

Interestingly, if we replace the probability distribution $\mathrm{P}(\cdot)$ by the optimal distribution $\mathrm{P}^*(\cdot)$ provided by Equation (3) in the objective function (2), we obtain for the minimum **free energy** between node 1 and node n,

$$\begin{aligned}
\phi_1^*(T) = \phi(\mathrm{P}^*) &= \sum_{\wp\in\mathcal{P}} \mathrm{P}^*(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}} \mathrm{P}^*(\wp)\log\left(\frac{\mathrm{P}^*(\wp)}{\tilde{\pi}(\wp)}\right) \\
&= \sum_{\wp\in\mathcal{P}} \mathrm{P}^*(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}} \mathrm{P}^*(\wp)\left(-\frac{1}{T}\tilde{c}(\wp) - \log\mathcal{Z}\right) \\
&= -T\log\mathcal{Z}
\end{aligned} \tag{4}$$

2.4 Computing quantities of interest from the partition function

Moreover, several quantities of interest can be computed by taking the partial derivative of the optimal free energy (4) [**?**, **?**, **?**, **?**, **?**, **?**].

Expected number of visits to edges and nodes. For instance, for the expected number of passages through edge (i,j) at temperature $T = 1/\theta$, that is, the flow in (i,j),

$$\begin{aligned}
\frac{\partial\phi(\mathrm{P}^*)}{\partial c_{ij}} &= -\frac{1}{\theta\mathcal{Z}}\frac{\partial\mathcal{Z}}{\partial c_{ij}} = -\frac{1}{\theta\mathcal{Z}}\sum_{\wp\in\mathcal{P}} \tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)](-\theta)\frac{\partial\tilde{c}(\wp)}{\partial c_{ij}} \\
&= \sum_{\wp\in\mathcal{P}} \frac{\tilde{\pi}(\wp)\exp[-\theta\tilde{c}(\wp)]}{\mathcal{Z}}\frac{\partial\tilde{c}(\wp)}{\partial c_{ij}} \\
&= \sum_{\wp\in\mathcal{P}} \mathrm{P}^*(\wp)\,\eta\big((i,j)\in\wp\big) \\
&= \bar{n}_{ij}(T)
\end{aligned} \tag{5}$$

where we used $\partial\tilde{c}(\wp)/\partial c_{ij} = \eta\big((i,j)\in\wp\big)$, with $\eta\big((i,j)\in\wp\big)$ being the number of times edge (i,j) appears on path $\wp$ at temperature T. Therefore, we have for the flow in (i,j),

$$\bar{n}_{ij}(T) = -T\frac{\partial\log\mathcal{Z}}{\partial c_{ij}} \tag{6}$$

Now, it turns out that the partition function can be computed in closed form (see, e.g., [**?**, **?**, **?**] for details). Let us first introduce the **fundamental matrix** of the RSP system,

$$\mathbf{Z} = \mathbf{I} + \mathbf{W} + \mathbf{W}^2 + \cdots = (\mathbf{I} - \mathbf{W})^{-1}, \quad \text{with } \mathbf{W} = \mathbf{P}_{\text{ref}} \circ \exp[-\theta \mathbf{C}] \tag{7}$$

where $\mathbf{C}$ is the cost matrix and $\circ$ is the elementwise (Hadamard) product. Elementwise, the entries of the $\mathbf{W}$ matrix are $w_{ij} = [\mathbf{W}]_{ij} = p_{ij}^{\text{ref}} \exp[-\theta c_{ij}]$. Note that this matrix is sub-stochastic because the costs are non-negative and node n is absorbing and killing (row n contains only 0 values).

Then, the partition function is simply $\mathcal{Z} = [\mathbf{Z}]_{1n} = z_{1n}$ (see [**?**, **?**, **?**, **?**, **?**]). More generally [**?**], we find that

$$z_{1k} = \sum_{\wp \in \mathcal{P}_{1k}} \tilde{\pi}(\wp) \exp[-\theta \tilde{c}(\wp)] \quad \text{and} \quad z_{kn} = \sum_{\wp \in \mathcal{P}_{kn}} \tilde{\pi}(\wp) \exp[-\theta \tilde{c}(\wp)] \tag{8}$$

with $z_{nn} = 1$, and where $\mathcal{P}_{1k}$ is the set of hitting paths starting in node 1 and ending in node k. Symmetrically, $\mathcal{P}_{kn}$ is the set of hitting paths starting in node k and ending in node n. The z_{1k} quantities are called the **forward variables** whereas the z_{kn} are the **backward variables**. These variables can be interpreted as probabilities of surviving during the killed random walk (see, e.g., [**?**, **?**] for details).

Moreover, the flow in (i, j) can be obtained from (7) by

$$\bar{n}_{ij}(T) = -\frac{1}{\theta} \frac{\partial \log \mathcal{Z}}{\partial c_{ij}} = \frac{z_{1i} p_{ij}^{\text{ref}} \exp[-\theta c_{ij}] z_{jn}}{z_{1n}} = \frac{z_{1i} w_{ij} z_{jn}}{z_{1n}} \tag{9}$$

and because only the first row and the last column of $\mathbf{Z}$ are needed, two systems of linear equations can be solved instead of matrix inversion in Equation (7).

From the last equation and $z_{in} = \sum_{j=1}^{n} w_{ij} z_{jn} + \delta_{in}$ (the elementwise form of $(\mathbf{I} - \mathbf{W})\mathbf{Z} = \mathbf{I}$), the expected number of visits to a node j can be computed from

$$\bar{n}_i(T) = \sum_{j=1}^{n} \bar{n}_{ij}(T) + \delta_{in} = \frac{z_{1i} z_{in}}{z_{1n}} \quad \text{for } i \neq n \tag{10}$$

where we assume $i \neq n$ for the last equality because we already know that $\bar{n}_n(T) = 1$ at the ending node, which is absorbing and killing.

Note that it can easily be shown that the relative entropy of the paths in Equation (2) can be rewritten in function of the expected number of visits and the *local* relative entropy as $\sum_{i=1}^{n-1} \bar{n}_i(T) \sum_{j \in \mathcal{S}ucc(i)} p_{ij} \log \frac{p_{ij}}{p_{ij}^{\text{ref}}}$ where the p_{ij} are the transition probabilities defining the current local policy of the random walker (probability of following a link) [**?**]. In fact, all the quantities of interest can be computed from the expected number of visits (see [**?**], Equation (11)).

Randomized, mixed, policy. Furthermore, the optimal transition probabilities of following an edge (i, j) with $i \neq n$ are

$$p_{ij}^*(T) = \frac{\bar{n}_{ij}(T)}{\bar{n}_i(T)} = p_{ij}^{\text{ref}} \exp[-\theta c_{ij}] \frac{z_{jn}}{z_{in}} = \frac{w_{ij} z_{jn}}{z_{in}} = \frac{w_{ij} z_{jn}}{\sum_{j' = \mathcal{S}ucc(i)} w_{ij'} z_{j'n}} \tag{11}$$

as $p_{ij}^{\text{ref}} \exp[-\theta c_{ij}] = w_{ij}$ and $z_{in} = \sum_{j = \mathcal{S}ucc(i)} w_{ij} z_{jn}$ for all $i \neq n$ (the elementwise form of $(\mathbf{I} - \mathbf{W})\mathbf{Z} = \mathbf{I}$, coming from Equation (7)). This expression defines a biased random walk on G – the random walker is "attracted" by the destination node n. These transition probabilities do not depend on the source node and correspond to the optimal randomized strategy, or policy, minimizing free energy. This randomized policy will be called a **mixed policy** as in game theory (see, e.g., [?]).

Expected cost until destination. In addition, the expected cost until reaching hitting node n is [?, ?, ?]

$$\langle \tilde{c} \rangle = \sum_{\wp \in \mathcal{P}} \mathrm{P}^*(\wp)\tilde{c}(\wp) = \sum_{\wp \in \mathcal{P}} \frac{\tilde{\pi}(\wp)\exp[-\theta \tilde{c}(\wp)]}{\mathcal{Z}} \tilde{c}(\wp) \tag{12}$$

By defining the matrix containing the expected number of passages through the edges by $\mathbf{N}$ with $[\mathbf{N}]_{ij} = \bar{n}_{ij}(T)$, the expected cost spread in the network is

$$\langle \tilde{c} \rangle = -\frac{\partial \log \mathcal{Z}}{\partial \theta} = \mathbf{e}^{\mathrm{T}}(\mathbf{N} \circ \mathbf{C})\mathbf{e} \tag{13}$$

where $\mathbf{e}$ is a column vector of 1s. This is just the cumulative sum of the expected number of passages through each edge times the cost of following the edge, $\sum_{i=1}^{n-1} \sum_{j \in \mathcal{S}ucc(i)} \bar{n}_{ij}(T) c_{ij}$ [?].

Entropy of the paths. In Equation (2), the relative entropy of the set of paths was defined as

$$J(\mathrm{P}^*|\tilde{\pi}) = \sum_{\wp \in \mathcal{P}} \mathrm{P}^*(\wp) \log\left(\frac{\mathrm{P}^*(\wp)}{\tilde{\pi}(\wp)}\right) \tag{14}$$

and, from Equations (2) and (4), it can be computed thanks to

$$J(\mathrm{P}^*|\tilde{\pi}) = -(\log \mathcal{Z} + \tfrac{1}{T}\langle \tilde{c} \rangle) \tag{15}$$

where the partition function $\mathcal{Z} = [\mathbf{Z}]_{1n} = z_{1n}$.

Free energy distance. It was already shown that the minimal free energy (4) at temperature T is provided by $\phi_1^*(T) = \phi(\mathrm{P}^*) = -T \log \mathcal{Z} = -\frac{1}{\theta} \log z_{1n}$. In [?, ?], it was proved that the free energy from any starting node i to absorbing node n, $\phi_i^*(T) = -\frac{1}{\theta} \log z_{in}$, can be computed thanks to the following recurrence formula to be iterated until convergence

$$\phi_i^*(T) = -\tfrac{1}{\theta} \log z_{in} = \begin{cases} -\frac{1}{\theta} \log \left[\displaystyle\sum_{j \in \mathcal{S}ucc(i)} p_{ij}^{\mathrm{ref}} \exp[-\theta(c_{ij} + \phi_j^*(T))] \right] & \text{if } i \neq n \\ 0 & \text{if } i = n \end{cases} \tag{16}$$

This equation is an extension of Bellman-Ford's formula for computing the shortest-path distance in a graph (see, e.g., [?, ?, ?, ?, ?, ?]). Moreover, the recurrence expression (16) is also a generalization of the distributed consensus algorithm developed in [?], considering binary costs only.

It was shown [?, ?] that this minimal free energy interpolates between the least cost ($T = \theta^{-1} \to \infty$; $\phi_i^*(\infty) = \min_{j \in \mathcal{S}ucc(i)}\{c_{ij} + \phi_j^*(\infty)\}$ and $\phi_n^*(\infty) = 0$)

and the expected cost before absorption ($T = \theta^{-1} \to 0$; $\phi_i^*(0) = \sum_{j \in \mathcal{S}ucc(i)} p_{ij}^{\text{ref}}(c_{ij} + \phi_j^*(0))$ and $\phi_n^*(0) = 0$) [?, ?, ?]. In addition [?, ?, ?], this quantity defines a **directed distance** between any node and absorbing node n.

In fact, as discussed in [?, ?], this last expression is obtained by simply replacing the min operator by a weighted version of the **softmin operator** ([?]; also called the Log-Sum-Exp function [?, ?]) in the Bellman-Ford recurrence formula,

$$\text{softmin}_{\mathbf{q},\theta}(\mathbf{x}) = -\frac{1}{\theta} \log \left(\sum_{j=1}^{n} q_j \exp[-\theta x_j] \right) \text{ with all } q_j \geq 0 \text{ and } \textstyle\sum_{j=1}^{n} q_j = 1 \tag{17}$$

which is a smooth approximation of the min operator and interpolates between weighted average and minimum operators, depending on the parameter θ [?, ?]. It also appeared in control [?] and exploration strategies for which an additional Kullback-Leibler cost term is incorporated in the immediate cost [?, ?, ?, ?, ?, ?, ?]. Moreover, this function was recently proposed as an operator guiding exploration in reinforcement learning, and more specifically for the SARSA algorithm [?, ?].

Note that the optimal mixed policy derived in Equation (11) can be rewritten in function of the free energy as

$$p_{ij}^*(T) = \frac{p_{ij}^{\text{ref}} \exp[-\theta c_{ij}] z_{jn}}{\sum_{j'=1}^{n} p_{ij'}^{\text{ref}} \exp[-\theta c_{ij'}] z_{j'n}} = \frac{p_{ij}^{\text{ref}} \exp[-\theta(c_{ij} + \phi_j^*(T))]}{\sum_{j'=1}^{n} p_{ij'}^{\text{ref}} \exp[-\theta(c_{ij'} + \phi_{j'}^*(T))]} \tag{18}$$

because $z_{in} = \exp[-\theta \phi_i^*(T)]$ and $z_{in} = \sum_{j=1}^{n} w_{ij} z_{jn} = \sum_{j=1}^{n} p_{ij}^{\text{ref}} \exp[-\theta c_{ij}] z_{jn}$ for all $i \neq n$. This corresponds to a multinomial logistic function.

3 Randomized shortest paths with constrained transition probabilities

Interestingly, the randomized shortest path formulation can easily be extended to account for some types of constraints. The goal here will be to determine the best mixed policy for reaching destination node n from source node 1 subject to *equality constraints* on some transition probabilities, given by Equation (18). We proceed as in previous section, but we now constrain the transition probabilities associated to some nodes to be equal to predefined values provided by the user. In other words, we constrain the relative flow passing through the edges starting from some nodes belonging to a set of nodes $\mathcal{C}$.

3.1 The Lagrange function

More precisely, the considered constraints on the nodes $i \in \mathcal{C}$ are

$$p_{ij}^*(T) = \frac{\bar{n}_{ij}(T)}{\bar{n}_i(T)} = \frac{\bar{n}_{ij}(T)}{\sum_{j \in \mathcal{S}ucc(i)} \bar{n}_{ij}(T)} = q_{ij} \text{ for the edges starting in nodes } i \in \mathcal{C} \tag{19}$$

which are independent of T. These transition probabilities q_{ij} are specified by the user for all the nodes in $\mathcal{C}$. We assume that these constraints are feasible. In particular, we must have $\sum_{j \in \mathcal{S}ucc(i)} q_{ij} = 1$ for all $i \in \mathcal{C}$. Moreover, the RSP model (see Equation (2)) implies that we should recover a pure random walk behavior, with reference probabilities provided by Equation (1), when $T \to \infty$. Therefore, to be consistent, the reference probabilities must be chosen such that $p_{ij}^{\text{ref}} = p_{ij}^{*}(T=\infty) = q_{ij}$ for nodes $i \in \mathcal{C}$. We will assume that this is the case in the sequel.

We consider the following Lagrange function

$$\begin{aligned}\mathscr{L}(\mathrm{P},\boldsymbol{\lambda}) = {} & \underbrace{\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\tilde{c}(\wp) + T\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)}_{\text{free energy, } \phi(\mathrm{P})} + \mu\left(\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) - 1\right) \\ & + \sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{S}ucc(i)} \lambda_{ij} \left[\underbrace{\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\,\eta\big((i,j)\in\wp\big)}_{\bar{n}_{ij}(T)} - q_{ij} \underbrace{\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\,\eta(i\in\wp)}_{\bar{n}_{i}(T)} \right]\end{aligned} \tag{20}$$

where, as before, $\mathcal{P}$ is the set of paths from node 1 to node n and with $\eta\big((i,j)\in\wp\big)$ being the number of times edge (i,j) appears on path $\wp$.

The Lagrange function can be rearranged as

$$\begin{aligned}\mathscr{L}(\mathrm{P},\boldsymbol{\lambda}) &= \sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) \underbrace{\left[\tilde{c}(\wp) + \sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{S}ucc(i)} \lambda_{ij}\,\eta\big((i,j)\in\wp\big) - \sum_{i\in\mathcal{C}} \eta(i\in\wp) \sum_{j'\in\mathcal{S}ucc(i)} q_{ij'}\,\lambda_{ij'} \right]}_{\tilde{c}'(\wp)} \\ &\quad + T\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right) + \mu\left(\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) - 1\right) \\ &= \underbrace{\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\tilde{c}'(\wp) + T\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp)\log\left(\frac{\mathrm{P}(\wp)}{\tilde{\pi}(\wp)}\right)}_{\text{free energy } \phi'(\mathrm{P})} + \mu\left(\sum_{\wp\in\mathcal{P}} \mathrm{P}(\wp) - 1\right)\end{aligned} \tag{21}$$

where, by using $\eta(i\in\wp) = \sum_{j\in\mathcal{S}ucc(i)} \eta\big((i,j)\in\wp\big)$, the local costs c_{ij} are redefined as

$$c'_{ij} = \begin{cases} c_{ij} + \underbrace{\lambda_{ij} - \sum_{j'\in\mathcal{S}ucc(i)} q_{ij'}\lambda_{ij'}}_{\text{cost update } \varDelta_{ij}} & \text{when node } i\in\mathcal{C} \\ c_{ij} & \text{otherwise} \end{cases} \tag{22}$$

and $\mathbf{C}'$ will be the matrix containing these new costs $c'_{ij} = c_{ij} + \varDelta_{ij}$, which will be called the *augmented* costs. We observe that Equation (21) is exactly a randomized shortest paths problem (see Equation (2)) containing augmented costs instead of the initial costs.

We further observe that the weighted (by transition probabilities) mean of the cost updates $\varDelta_{ij} = \lambda_{ij} - \sum_{j'\in\mathcal{S}ucc(i)} q_{ij'}\lambda_{ij'}$ is equal to zero on each node $i\in\mathcal{C}$: $\sum_{j\in\mathcal{S}ucc(i)} q_{ij}\varDelta_{ij} = 0$. This implies that the weighted average of the augmented costs is equal to the weighted average of the original costs on each constrained node i, $\sum_{j\in\mathcal{S}ucc(i)} q_{ij}c'_{ij} = \sum_{j\in\mathcal{S}ucc(i)} q_{ij}c_{ij}$. Therefore, this choice ensures that the expected augmented cost is equal to the real expected cost provided by Equation (13), $\langle \tilde{c}' \rangle = \mathbf{e}^{\mathrm{T}}(\mathbf{N} \circ \mathbf{C}')\mathbf{e} = \langle \tilde{c} \rangle$. In this case, the perceived cost when visiting any node using the augmented costs ($\mathbf{C}'$) is exactly the same in average as the perceived real cost ($\mathbf{C}$) when no constraint is introduced.

Thus, in Equation (21), everything happens as if the costs have been redefined by taking into account the Lagrange parameters. These Lagrange parameters can therefore be interpreted as additional costs necessary to satisfy the equality constraints. We now have to find the λ_i by Lagrangian duality.

Let $\phi'(\mathrm{P})$ be the free energy obtained from these augmented costs. We now have to find the λ_{ij} by Lagrangian duality.

3.2 Exploiting Lagrangian duality

In this section, we will take advantage of the fact that, in our formulation of the problem, the Lagrange dual function and its gradient are easy to compute.

As the objective function is convex and all the equality constraints are linear, there is only one global minimum and the duality gap is zero [**?**, **?**, **?**]. The optimum is a saddle point of the Lagrange function and a common optimization procedure (often called Arrow-Hurwicz-Uzawa [**?**]) consists in sequentially (i) solving the primal while considering the Lagrange parameters as fixed and then (ii) solving the dual (which is concave) while considering the variables to be optimized (the paths probabilities) as fixed, until convergence.

In our context, this provides the two following steps which are iterated

$$\begin{cases} \mathscr{L}(\mathrm{P}^*, \boldsymbol{\lambda}) = \underset{\{\mathrm{P}(\wp)\}_{\wp \in \mathcal{P}}}{\text{minimize}}\, \mathscr{L}(\mathrm{P}, \boldsymbol{\lambda}) & \text{(compute the dual function)} \\ \mathscr{L}(\mathrm{P}^*, \boldsymbol{\lambda}^*) = \underset{\boldsymbol{\lambda}}{\text{maximize}}\, \mathscr{L}(\mathrm{P}^*, \boldsymbol{\lambda}) & \text{(maximize the dual function)} \end{cases} \tag{23}$$

and this is the procedure that will be followed, where the dual function will be maximized through a simple coordinate ascend on Lagrange multipliers.

3.3 Computing the dual function

We already know from (3) that the first step in Equation (23) leads to

$$\mathrm{P}^*(\wp) = \frac{\tilde{\pi}(\wp) \exp[-\theta \tilde{c}'(\wp)]}{\displaystyle\sum_{\wp' \in \mathcal{P}} \tilde{\pi}(\wp') \exp[-\theta \tilde{c}'(\wp')]} = \frac{\tilde{\pi}(\wp) \exp[-\theta \tilde{c}'(\wp)]}{\mathcal{Z}'} \tag{24}$$

where $\tilde{c}'(\wp)$ is the augmented cost of a path. Notice that once the optimal probability distribution on paths has been computed, the expected number of transitions through any edge can be deduced from Equation (5) and the transition probabilities from (11). The partition function $\mathcal{Z}'$, and the z'_{in} in general, can then be obtained from Equation (11).

Then, from Equations (4) and (21), the dual function can easily be computed in function of the partition function defined from the augmented costs,

$$\mathscr{L}(\mathrm{P}^*, \boldsymbol{\lambda}) = -T \log \mathcal{Z}' \tag{25}$$

and has to be maximized with respect to the $\{\lambda_{ij}\}$ with $i \in \mathcal{C}$ and $j \in \mathcal{S}ucc(i)$.

3.4 Maximizing the dual function

Let us now try to maximize the dual function. Because $\bar{n}_i(T) = \sum_{j \in \mathcal{S}ucc(i)} \bar{n}_{ij}(T)$, by following the reasoning of previous subsection, we obtain

$$\begin{aligned}
\frac{\partial \mathscr{L}(\mathrm{P}^*, \boldsymbol{\lambda})}{\partial \lambda_{ij}} &= \frac{\partial(-T \log \mathcal{Z}')}{\partial \lambda_{ij}} \\
&= \sum_{j' \in \mathcal{S}ucc(i)} \frac{\partial(-T \log \mathcal{Z}')}{\partial c'_{ij'}} \frac{\partial c'_{ij'}}{\partial \lambda_{ij}} \\
&= \sum_{j' \in \mathcal{S}ucc(i)} \bar{n}_{ij'}(T)(\delta_{jj'} - q_{ij}) \\
&= \bar{n}_{ij}(T) - q_{ij}\bar{n}_i(T) \qquad (26)
\end{aligned}$$

Quite naturally, setting the result to zero provides the constraints for nodes $i \in \mathcal{C}$,

$$\frac{\bar{n}_{ij}(T)}{\bar{n}_i(T)} = q_{ij} \qquad (27)$$

and we now have to solve these equations in terms of the Lagrange parameters, λ_{ij}.

3.5 Computing the Lagrange parameters

Recalling that $\bar{n}_i(T) = \sum_{j \in \mathcal{S}ucc(i)} \bar{n}_{ij}(T)$ and Equations (9) and (11), we obtain, for each $i \in \mathcal{C}$ and $j \in \mathcal{S}ucc(i)$,

$$\frac{p_{ij}^{\mathrm{ref}} \exp[-\theta c'_{ij}] z_{jn}}{z_{in}} = q_{ij} \qquad (28)$$

The objective of this subsection is to compute each augmented cost (and thus the Lagrange parameter λ_{ij}, see Equation (22)) corresponding to nodes $i \in \mathcal{C}$ from this equation by isolating c'_{ij}, given that the backward variables z_{in} are fixed once we know the optimal path distribution.

Recall (see the discussion following Equation (19)) that it was assumed that the imposed q_{ij} and the reference transition probabilities p_{ij}^{ref} are consistent on constrained nodes, that is, $p_{ij}^{\mathrm{ref}} = q_{ij}$ for all $i \in \mathcal{C}$. We thus obtain, for $i \in \mathcal{C}$,

$$\frac{\exp[-\theta c'_{ij}] z_{jn}}{z_{in}} = \frac{\exp[-\theta c'_{ij}] z_{jn}}{\sum_{j' \in \mathcal{S}ucc(i)} p_{ij'}^{\mathrm{ref}} \exp[-\theta c'_{ij'}] z_{j'n}} = 1 \qquad (29)$$

as $z_{in} = \sum_{j=1}^{n} p_{ij}^{\mathrm{ref}} \exp[-\theta c_{ij}] z_{jn}$ for all $i \neq n$ (see Equation (18)).

This shows that the augmented costs related to a constrained node are only defined up to the addition of a constant term (a translation). This will give us the opportunity to constrain the weighted average of the augmented costs $c'_{ij} = c_{ij} + \Delta_{ij}$ to be equal to the weighted average of the original costs, as required – see the discussion following Equation (22). The Δ_{ij} have to be computed from this previous equation while satisfying this constraint, which reduces to $\Delta_{ij} = \lambda_{ij} - \sum_{j' \in \mathcal{S}ucc(i)} q_{ij'} \lambda_{ij'}$, as shown before.

Let us isolate Δ_{ij} in the previous equation:

$$\exp[\theta \Delta_{ij}] = \frac{\exp[-\theta c_{ij}] z_{jn}}{z_{in}} \qquad (30)$$

Then, by taking $\frac{1}{\theta}\log$ of both sides,

$$\begin{aligned}\varDelta_{ij} &= \tfrac{1}{\theta}\log(\exp[-\theta c_{ij}]z_{jn}) - \tfrac{1}{\theta}\log z_{in} \\ &= \underbrace{\tfrac{1}{\theta}\log(z_{jn}) - c_{ij}}_{\lambda_{ij}} - \tfrac{1}{\theta}\log z_{in}\end{aligned} \tag{31}$$

where we defined $\lambda_{ij} = \frac{1}{\theta}\log(z_{jn}) - c_{ij}$. Multiplying both sides by q_{ij} and summing over j provides

$$\tfrac{1}{\theta}\log z_{in} = \sum_{j\in\mathcal{S}ucc(i)} q_{ij}\big(\tfrac{1}{\theta}\log(z_{jn}) - c_{ij}\big) \tag{32}$$

which, from (31), just aims to center the λ_{ij}, as required.

This suggests the following sequential steps to be applied on each node $i \in \mathcal{C}$ in turn

$$\begin{cases}\lambda_{ij} \leftarrow \frac{1}{\theta}\log(z_{jn}) - c_{ij} & \text{for all } j \in \mathcal{S}ucc(i) \\ \varDelta_{ij} \leftarrow \lambda_{ij} - \sum_{j'\in\mathcal{S}ucc(i)} q_{ij'}\lambda_{ij'} & \text{for all } j \in \mathcal{S}ucc(i) \\ c'_{ij} \leftarrow c_{ij} + \varDelta_{ij} & \text{for all } j \in \mathcal{S}ucc(i)\end{cases} \tag{33}$$

until convergence. Recall also that, for consistency, $q_{ij} = p^{\text{ref}}_{ij}$ on constrained nodes.

3.6 The final procedure

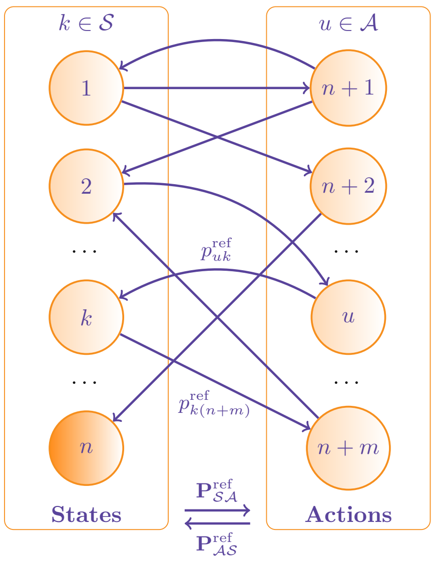
Therefore, after initializing the Lagrange parameters to 0, the final procedure iterates the following steps:

- ▶ The elements of the fundamental matrix are computed thanks to Equation (7) from the augmented costs c'_{ij} (defined in Equation (22)) and from the transition matrix of the natural random walk on G (Equation (1)), where destination node n is made absorbing and killing.
- ▶ The augmented costs are updated for all edges incident to constrained nodes (in $\mathcal{C}$) thanks to the three cases detailed in Equation (33).

After convergence of the two previous steps, the optimal policy is computed thanks to Equation (11) by using the augmented costs c'_{ij} instead of c_{ij}. This provides optimal transition probabilities $p^*_{ij}(T)$. We now apply this procedure in order to solve simple Markov decision problems.

4 Markov decision processes as a constrained randomized shortest path on a bipartite graph

The previous sections described all the needed tools for computing an optimal mixed policy on a Markov decision process (MDP). Recall that, as in [?], we assume that there is a special cost-free **destination** or **goal** node n; once the system has reached that node, it simply disappears (killing node). Thus, node n has no outgoing link. As in [?], we will also consider a problem structure such that termination is inevitable. Thus, the horizon is in effect finite, but its length is random and it depends on the policy being used. The conditions for



$k \in \mathcal{S}$
$u \in \mathcal{A}$
1
2
k
n
$n+1$
$n+2$
u
$n+m$
p^{ref}_{uk}
$p^{\mathrm{ref}}_{k(n+m)}$
States
Actions
$\mathbf{P}^{\mathrm{ref}}_{\mathcal{SA}}$
$\mathbf{P}^{\mathrm{ref}}_{\mathcal{AS}}$

Figure 1: The bipartite graph with states on the left side ($\mathcal{S}$) and control actions on the right ($\mathcal{A}$). Node 1 is the initial state while node n is the absorbing, destination, state of the system. The transitions probabilities from states to actions p^{ref}_{ku} are gathered in matrix $\mathbf{P}^{\mathrm{ref}}_{\mathcal{SA}}$ and the transition probabilities from actions to states p^{ref}_{ku}, provided by the environment, are gathered in matrix $\mathbf{P}^{\mathrm{ref}}_{\mathcal{AS}}$.

which this is true are, basically, related to the fact that the destination node can be reached in a finite number of steps from any potential initial node; for a rigorous treatment, see e.g. [**?**, **?**].

The main objective is thus to design a **randomized**, **mixed**, **policy** minimizing the expected cost-to-go subject to an entropy constraint controlling the total randomness spread in the network, and therefore the exploration rate. In other words, we are looking for an optimal policy or, in our case, an optimal transition probabilities matrix $\mathbf{P}^{*}$ of a finite, first-order, Markov chain minimizing the expected cost needed to reach the destination state from the initial state, while fixing the entropy spread in the chain as well as the transition probabilities provided by the environment.

Therefore, the solution is obtained by the algorithm described in Subsection 3.6 – the randomized shortest paths with constraints on transition probabilities – applied to a bipartite graph, as described now.

4.1 Basic framework and procedure

The Markov decision process is now viewed as a (constrained) randomized shortest path on a bipartite graph (see Figure 1). First, we examine how the reference transition probabilities defining the natural random walk on this graph are defined. Then, the structure of the bipartite graph and the way to compute the optimal policy are described. Finally, the precise form of the matrices needed to run the constrained RSP is detailed.

Pure random walk: reference probabilities. More precisely, in the case of a *pure random walk* (the reference transition probabilities (Equation 1), corresponding to $T \to \infty$ in Equation (2)), we consider that agents are sent from an initial state 1 and that, at each state $s = k$ (n states in total, $\mathcal{S}$), they choose an action $a_k = u$ with a probability mass p^{ref}_{ku}, $k \in \mathcal{S}$ and $u \in \mathcal{U}(k)$, the set of actions available in state k. When no prior information on the system is available, these are usually set to $p^{\text{ref}}_{ku} = 1/|\mathcal{U}(k)|$ (a uniform distribution). In our bipartite graph, $\mathcal{U}(k)$ is nothing else than the successors of node k, $\mathcal{U}(k) = \mathcal{S}ucc(k)$.

Agents then perform the chosen action u, and incur a finite cost c_{ku} associated to the execution of action u in state k. They then jump to the next state $s = l$ with a reference transition probability p^{ref}_{ul} provided by the environment, where $l \in \mathcal{S}$ and $u \in \mathcal{A}$, depending on the action. This behavior corresponds to a pure exploration, according to the Markov chain with transition probabilities p^{ref}_{ul} and, for consistency, these transition probabilities must be equal to the constrained transition probabilities, q_{ul}, as discussed in the previous section. Notice that an additional cost could also be associated to the transition to state l, after action u is performed, as, e.g., in [**?**], but in this work we will adopt the simpler setting where the cost is a function of the action u in state k only [**?**, **?**].

Definition of the bipartite graph. Therefore, the process can be modeled as a directed **bipartite graph** G_{b} in which the left nodes are the original states $\mathcal{S}$ and the right nodes are the possible actions associated to the states, $\mathcal{A} = \mathcal{U}(1) \cup \mathcal{U}(2) \cup \ldots \cup \mathcal{U}(n-1)$ (n is absorbing and has no associated action). We have $n = |\mathcal{S}|$ and $m = |\mathcal{A}|$. Note that each action associated to a state is a node of G_{b}, even if the same action is also available in some other nodes – actions are duplicated for each node in which they appear. Therefore, the number of such right nodes is $|\mathcal{A}| = |\mathcal{U}(1)| + |\mathcal{U}(2)| + \cdots + |\mathcal{U}(n-1)| = m$.

Moreover, it is assumed that, in this bipartite graph G_{b}, the nodes corresponding to states $\mathcal{S}$ are numbered first (from 1 to n) and actions $\mathcal{A}$ are following (from $n+1$ to $n+m$). Therefore, the set of available actions in any state k is nothing else that the successor nodes of k, $\mathcal{U}(k) = \mathcal{S}ucc(k)$.

The optimal mixed policy. When the temperature T decreases, the agents are more and more exploiting good policies while still exploring the environment – they interpolate between a purely random behavior (guided by the reference probabilities) and the best, deterministic, policy solving the Markov decision process, provided, e.g., by the value iteration algorithm [**?**, **?**, **?**, **?**]. The control actions $u \in \mathcal{A}$ are then chosen according to an optimal stochastic **policy** Π^*, mapping every state k to the set $\mathcal{U}(k)$ of available actions with a probability mass $p^*_{ku}(T)$ provided by the randomized shortest path model applied on graph G_{b} (see Equation (11)). The policy $\Pi \equiv \{p^*_{ku}(T), k \in \mathcal{S} \wedge u \in \mathcal{U}(k)\}$ defines, for each state k, an optimal probability distribution on the set $U(k)$ of actions available in this state, provided by Equation (11), and gradually biasing the walk towards the optimal, deterministic, policy. The policy is optimal in the sense that it minimizes expected cost for a given degree of relative entropy (see Equation (2)).

For instance, if the available actions in some state k are $\mathcal{U}(k) = \{5, 6, 7\}$, the probability mass $p^*_{ku}(T)$ specifies three probabilities $p^*_{k5}(T)$, $p^*_{k6}(T)$, and $p^*_{k7}(T)$, summing to one. These optimal transition probabilities are provided by the policy obtained from the constrained randomized shortest paths model applied on the bipartite graph G_{b}, that is, by Equation (11).

As discussed in [?], such random choices are common in a variety of fields, for instance decision sciences [?] or game theory, where they are called mixed strategies (see, e.g., [?]). Thus, the problem tackled in this section simply corresponds to a constrained **randomized shortest-path problem** (RSP) on G_{b}.

Useful matrices. Recall that, in the randomized shortest path framework, the procedure for computing the optimal policy takes three quantities in input: the *reference transition probabilities*, the *cost matrix* and the *set of constrained nodes*.

The $n \times m$ reference transition probabilities matrix $\mathbf{P}^{\mathrm{ref}}_{\mathcal{SA}}$ (states-actions) contains the p^{ref}_{ku} for the transitions from the left nodes (states in $\mathcal{S}$) and the right nodes (actions in $\mathcal{A}$) of G_{b}. For node n (the killing absorbing node), the corresponding row of $\mathbf{P}^{\mathrm{ref}}_{\mathcal{SA}}$ (the last row) is set equal to 0. Indeed, there are no actions associated to this node.

In a symmetric way, once an action in $\mathcal{A}$ has been chosen, the agent is in the corresponding action node in the bipartite graph G_{b}, say node $u \in \mathcal{A}$. Then, he jumps to a state $k \in \mathcal{S}$ with probability p^{ref}_{uk} (actions-states), predefined by the environment. It reflects the probability of ending in state $s = k$ once action $a = u$ has been chosen. The corresponding $m \times n$ transition probabilities matrix from actions to states is $\mathbf{P}^{\mathrm{ref}}_{\mathcal{AS}}$ and its elements are equal to the constrained transition probabilities, $p^{\mathrm{ref}}_{uk} = q_{uk}$. The set of action nodes is therefore the set of constrained nodes.

Consequently, the transition matrix of the bipartite graph G_{b} is

$$\mathbf{P}^{\mathrm{ref}} = \begin{array}{c c} & \begin{array}{cc} \mathcal{S} & \mathcal{A} \end{array} \\ \begin{array}{c} \mathcal{S} \\ \mathcal{A} \end{array} & \left[\begin{array}{cc} \mathbf{O} & \mathbf{P}^{\mathrm{ref}}_{\mathcal{SA}} \\ \mathbf{P}^{\mathrm{ref}}_{\mathcal{AS}} & \mathbf{O} \end{array} \right] \end{array} \tag{34}$$

where $\mathbf{O}$ is a 0 matrix of the appropriate size. Its elements are p^{ref}_{ij}.

Moreover, the cost matrix for the bipartite graph G_{b} is

$$\mathbf{C}_{\mathrm{b}} = \begin{array}{c c} & \begin{array}{cc} \mathcal{S} & \mathcal{A} \end{array} \\ \begin{array}{c} \mathcal{S} \\ \mathcal{A} \end{array} & \left[\begin{array}{cc} \mathbf{O} & \mathbf{C} \\ \mathbf{O} & \mathbf{O} \end{array} \right] \end{array} \tag{35}$$

that is, as in [?, ?], costs are only defined on state-action edges, c_{ku} with $k \in \mathcal{S}$ and $u \in \mathcal{A}$. The other costs are equal to zero (no incurred cost for action-state transitions). Extensions to costs defined on action-state edges are possible, but are omitted to keep things simple. Row n of this matrix, corresponding to the killing absorbing node, is set to 0.

Finally, the set of constrained nodes is simply the set of action nodes $\mathcal{A}$ from which the transition probabilities are provided by the environment, $\mathbf{P}^{\mathrm{ref}}_{\mathcal{AS}}$.

Computing the optimal policy. Once these matrices are computed, the optimal randomized policy interpolating between the optimal deterministic policy of the standard MDP and a pure random walk defined by the reference transition matrix $\mathbf{P}^{\mathrm{ref}}$ is obtained by applying the randomized shortest path procedure on the graph G_{b} (see Subsection 3.6). The optimal mixed policy is obtained from Equation (11).

We now describe a simplified way for obtaining the optimal mixed policy, inspired by the value iteration algorithm.

4.2 A simple soft value iteration algorithm

Interestingly and surprisingly, we will now show that replacing the minimum operator by a softmin operator in the standard value iteration algorithm provides exactly the same results as the constrained RSP. This property will lead to a simple algorithm, extending the standard value iteration algorithm, for computing randomized policies.

This shows that the proposition of using the softmin function for exploration in reinforcement learning [?, ?, ?, ?, ?, ?, ?, ?, ?] is also globally optimal in that it minimizes expected path cost subject to a fixed relative entropy of paths equality constraint (see Equation (2)), at least in our setting of a absorbing, goal, node n reachable from any other node of the graph.

4.2.1 The value iteration algorithm

Let us first recall the **standard value iteration** procedure, computing the expected cost until absorption by the goal state n [?, ?, ?, ?] when starting from a node $k \in \mathcal{S}$, denoted by v_{kn}, and based on the following recurrence formula

$$v_{kn} = \begin{cases} \min\limits_{u \in \mathcal{U}(k)} \left\{ c_{ku} + \sum\limits_{l \in \mathcal{S}ucc(u)} p_{ul}^{\mathrm{ref}} v_{ln} \right\} & \text{if } k \neq n \\ 0 & \text{if } k = n \end{cases} \tag{36}$$

where p_{ul}^{ref} is element u, l (with $u \in \mathcal{A}$ and $l \in \mathcal{S}$) of matrix $\mathbf{P}_{\mathrm{ref}}$ of the reference random walk on the bipartite graph (see Equation (34)). This expression is iterated until convergence, which is guaranteed under some mild conditions, for any set of nonnegative initial values (see, e.g., [?, ?, ?, ?, ?] for details).

4.2.2 The soft value iteration algorithm

We start from the softmin form of the free energy ([?, ?, ?]; see also Equation (16)), which corresponds to a Bellman-Ford algorithm for computing the shortest path distance in which the min operator has been replaced by the softmin operator defined in Equation (17). Substituting in the same way the softmin operator for the min in the value iteration update formula provides a randomized equivalent of the Bellman-Ford optimality conditions,

$$\phi_k^{\mathcal{S}}(T) = \begin{cases} -\frac{1}{\theta} \log \left[\sum\limits_{u \in \mathcal{U}(k)} p_{ku}^{\mathrm{ref}} \exp \left[-\theta \Big(c_{ku} + \sum\limits_{l \in \mathcal{S}ucc(u)} p_{ul}^{\mathrm{ref}} \phi_l^{\mathcal{S}}(T) \Big) \right] \right] & \text{if } k \neq n \\ 0 & \text{if } k = n \end{cases} \tag{37}$$

which is similar to the expression computing the free energy in Equation (16) where the update is replaced by an equality. The quantity $\phi_k^{\mathcal{S}}(T) = -T \log z_{kn} =$

$-\frac{1}{\theta}\log z_{kn}$ (see Appendix A for details), where z_{kn} is the backward variable introduced in Equation (8), will be called the **free energy** of the Markov decision process, associated to the different states $k \in \mathcal{S}$. This equation states the necessary optimality conditions and will be called the **Bellman-Ford optimality conditions** for the randomized Markov decision process.

Note that it can be shown (see the appendix of [?, ?]) that this recurrence formula reduces to the standard optimality conditions for Markov decision processes (Equation (36)) when $\theta \to \infty$. Conversely, when $\theta \to 0$, it reduces to the expression allowing to compute the expected cost until absorption by the goal state n, also called the average first-passage cost [?, ?], $\phi_k^{\mathcal{S}}(T) = \sum_{u \in \mathcal{U}(k)} p_{ku}^{\text{ref}}(c_{ku} + \sum_{l \in \mathcal{S}ucc(u)} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T))$. Furthermore, it can be shown that the backward variables z_{in} can be interpreted as the probabilities of reaching the goal state n, and thus of surviving, during a killed random walk on G_{b} with transition probabilities $w'_{ij} = p_{ij}^{\text{ref}} \exp[-\theta c'_{ij}]$, for $i \in \mathcal{S}$, $j \in \mathcal{A}$, and vice-versa [?, ?, ?]. The quantity $\phi_k^{\mathcal{S}}(T)$ is the global cost (free energy) associated to this probability of surviving.

This suggests the use of the following soft form of value iteration for computing the solution of the randomized Markov decision system by replacing the equality by an update in (37),

$$\phi_k^{\mathcal{S}}(T) \leftarrow \begin{cases} -\frac{1}{\theta}\log\left[\displaystyle\sum_{u \in \mathcal{U}(k)} p_{ku}^{\text{ref}} \exp\left[-\theta\Big(c_{ku} + \sum_{l \in \mathcal{S}ucc(u)} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T)\Big)\right]\right] & \text{if } k \neq n \\ 0 & \text{if } k = n \end{cases} \tag{38}$$

which has to be iterated over all states until convergence.

After convergence of the values to a fixed point, optimality conditions (37) should be verified. Then, the optimal policy for each node $k \in \mathcal{S}$ and $k \neq n$ is computed thanks to

$$p_{ku}^{*}(T) = \frac{p_{ku}^{\text{ref}} \exp\left[-\theta\Big(c_{ku} + \displaystyle\sum_{l \in \mathcal{S}ucc(u)} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T)\Big)\right]}{\displaystyle\sum_{u' \in \mathcal{U}(k)} p_{ku'}^{\text{ref}} \exp\left[-\theta\Big(c_{ku'} + \sum_{l \in \mathcal{S}ucc(u')} p_{u'l}^{\text{ref}} \phi_l^{\mathcal{S}}(T)\Big)\right]} \quad \text{for } k \neq n \tag{39}$$

which provides the probability of choosing action u within state k.

This procedure, involving the iteration of Equation (37) and the computation of the optimal policy from Equation (39), will be called the **soft value iteration** algorithm.

As for the free energy of the bag-of-paths system [?, ?], we derive – and thus justify theoretically – this iterative algorithm from the randomized shortest paths framework. The derivation is provided in Appendix A.

Finally, in [?], it was shown that the iterative update of an expression similar (but simpler) to Equation (38) converges and is independent of the initial values. We prove the same property for the soft value iteration in Appendix B by using a fixed-point theorem point of view, implying that the update (38) is a contraction mapping. Besides theoretical convergence, we observed empirically in all our experiments that both techniques (the soft value iteration and the constrained randomized shortest path procedures) converge and provide exactly the same policies.

Table 1: Results of the soft value iteration method (SVI with a high θ) performed on the 421 game. The first column represents the states of the set of three dices (dices drawings are sorted in increasing order). The second column provides the corresponding reward scores. The third and fourth columns show the optimal strategies provided by the Markov decision process, respectively for the first re-roll and the second one (which dice has to be re-rolled).

DDD	Score	SVI reroll 1	SVI reroll 2	DDD	Score	SVI reroll 1	SVI reroll 2
111	7	0 0 0	0 0 0	543	2	1 0 1	0 0 0
211	2	1 0 0	1 0 0	544	1	1 0 1	1 0 1
221	1	0 1 0	0 1 0	551	1	1 1 0	1 1 0
222	3	0 0 0	0 0 0	552	1	1 1 1	0 0 1
311	3	1 0 0	1 0 0	553	1	1 1 1	0 0 1
321	2	1 0 0	1 0 0	554	3	1 1 0	1 1 0
322	1	1 1 1	1 1 1	555	6	0 0 0	0 0 0
331	1	1 1 0	1 1 0	611	1	0 0 0	0 0 0
332	1	1 1 1	1 1 1	621	1	1 0 0	1 0 0
333	3	0 0 0	0 0 0	622	1	1 1 1	1 1 1
411	4	1 0 0	1 0 0	631	1	1 1 0	1 1 0
421	8	0 0 0	0 0 0	632	1	1 1 1	1 1 1
422	1	0 0 1	0 0 1	633	1	1 1 1	1 1 1
431	1	0 1 0	0 1 0	641	1	1 0 0	1 0 0
432	2	0 1 0	0 1 0	642	1	1 0 0	1 0 0
433	1	0 1 1	0 1 1	643	1	1 0 1	1 0 1
441	1	0 1 0	0 1 0	644	1	1 0 1	1 0 1
442	1	0 1 0	0 1 0	651	1	1 1 0	1 1 0
443	1	1 0 1	1 0 1	652	1	1 1 1	1 1 1
444	3	0 0 0	0 0 0	653	1	1 1 1	1 1 1
511	5	1 0 0	0 0 0	654	2	1 1 0	0 0 0
521	1	1 0 0	1 0 0	655	1	1 1 1	1 0 0
522	1	1 1 1	1 1 1	661	1	1 1 0	1 1 0
531	1	1 1 0	1 1 0	662	1	0 0 1	0 0 1
532	1	1 1 1	1 1 1	663	1	0 0 1	0 0 1
533	1	1 1 1	1 1 1	664	1	0 0 1	0 0 1
541	1	1 0 0	1 0 0	665	1	0 0 1	0 0 1
542	1	1 0 0	1 0 0	666	3	0 0 0	0 0 0

5 Simulations and discussion: application to the 421 dice game

This section describes an experiment illustrating the application of constrained randomized shortest paths to Markov decision problems. Several different simulations have been run on four different problems but, in order to save space, we decided to report only one application, the 421 game.

5.1 Rules of the 421 game

The 421 dice game (quatre-cent-vingt-et-un in french) is popular in France and Belgium and is played with three six-faced dice and 11 tokens (representing penalties). A player looses the game when he gets all tokens – the goal is thus to get rid of our tokens. The game is composed of two phases:

- During the first phase, each player rolls all three dices (no re-rolls are allowed) and reward scores according to the second column on Table 1. Observe that after each full turn, the player with the *lowest resulting score* (according to Table 1) must take a certain number of tokens from the pot. This number is equal to the *highest resulting score* for this turn. The game

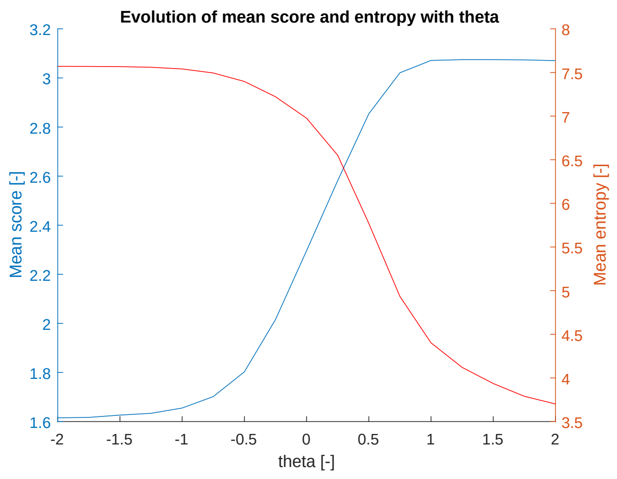


Figure 2: For an increasing θ (in logarithmic scale), all mixed strategies based on the soft value iteration (see Equation (38)) are comared and results are reported. The blue curve depicts the evolution of the average (over $10e6$ turns) score at the end of each turn (reward, larger is better) for an "artificial" player, whose policy is provided by the constrained randomized shortest path policy, in terms of the strategies obtained with different values of the θ parameter. Conversely, the red curve indicates the average entropy of the different strategies. The largest entropy is achieved when θ is smallest, and is minimum when θ is largest.

continues until the pot is empty. Therefore, the goal of the first phase is to distribute the 11 tokens among players. Thereafter, the second phase can start.

- During the second phase, the players also roll all three dices a first time, but are now allowed to further *re-roll them two times* (all or part of the three dices). Thus, after having rolled all three dices once, the player has the choice to re-roll dice number 1, number 2 or number 3, two of these, or even re-roll them all, and this two times. This means that he has two chances to enhance his score obtained after the first drawing. After that, he passes his turn to the next player. Therefore, in this game, the *decision actions* are how many dices to re-roll (0, 1, 2 or 3) and, in each of these cases, which dice to re-roll. Moreover, there are two sequential decisions since the player has the opportunity to re-roll the dices two times. Note that the *state* for the player corresponds to the combined state of the three dices (column 1 of Figure **??**). Based on this state, the player has to decide which action has to be performed. Then after re-rolling (or not) the dices, we end up in a new state and the same applies after the (potential) second re-roll.

  Furthermore, after each full turn, the player with the *lowest score* (the loser for this turn) must take a given number of tokens from the player that scored the *highest* (the winner for this turn). Here again, this number is equal to the *largest obtained score* during this turn. The game goes on until someone gets all the tokens (11). In this case, he looses the game (and must pay a beer to the players).

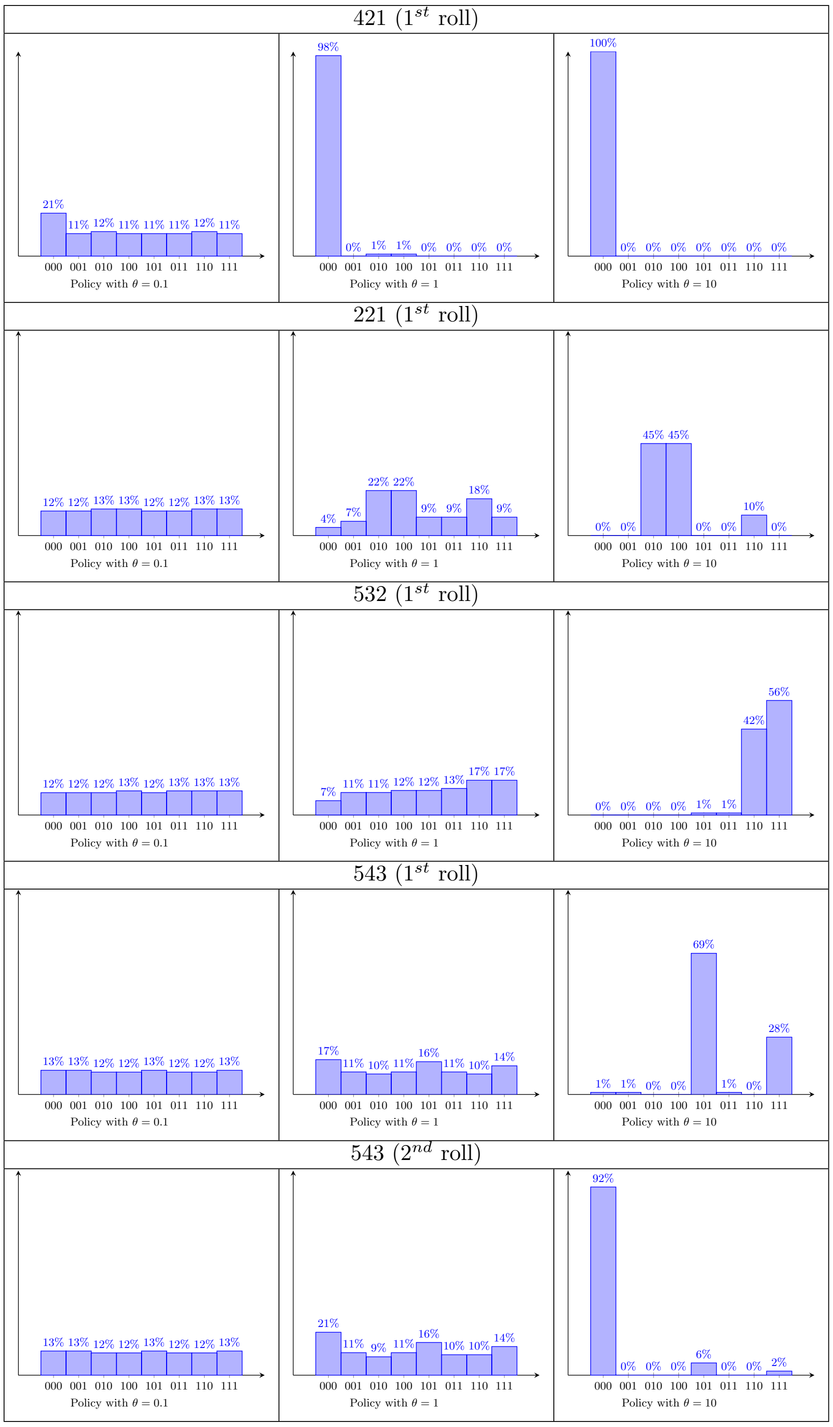


Figure 3: Mixed (randomized) policy provided by the soft value iteration (SVI, Equations (38)-(39)), for some interesting cases. The resulting policies interpolate between a uniform distribution (equal to the reference transition probabilities, left) when $\theta \to 0$ and the deterministic, optimal, policy obtained by the value iteration algorithm (sum of Kronecker delta, right), when $\theta \to \infty$.

Thus, the Markov decision process (MPD) must only be run during the second phase. In this example, we do not take into account the interactions among players, that is, the number of tokens of each player. The goal is to maximize the reward scores of the considered player, knowing that zero, one or two re-roll(s) are allowed and we can choose each time which dice to re-draw. There are 3*56 states for this game (the 56 states of column 1 of Figure **??** must be encoded three times: once for after the first roll, once for after the first re-roll and once for the second re-roll) plus a (virtual) starting state. Meanwhile, only 56 states from the $6 \times 6 \times 6$ permutations of the three dices are considered, as the order of the dices is not taken into account. In this 421 game, the *reward* at the end of each turn is defined as the number of tokens you get rid of (transferred to the looser) assuming that you are the winner.

5.2 Simulation results

The optimal policies obtained using the constrained randomized shortest paths framework (see Section 3.6) or using the soft value iteration algorithm (SVI, Equations (38)-(39)) are reported for a high θ on Table 1. We of course verified experimentally that the two procedures converge to exactly the same values and that both converge to the optimal, deterministic, policy when θ increases. Note that the states are denoted as follows. DDD in Table 1, column 1, indicates the results of the three dices ordered from the highest value to the lowest.

The optimal policy is then reported in columns 3 and 4 for the first and the second re-roll. Note that the strategy can be different during the first re-roll or the second re-roll (see, e.g., DDD = 543). Re-rolls are coded using three booleans (for instance, 011 means "re-roll dice 2 and 3 but not dice 1". The mean *reward*, defined as the obtained score after a turn, is 3.06 when playing the optimal policy. Conversely, when playing randomly and choosing the policy according to the reference probabilities, the mean reward is much lower, 1.61. It is important to have these values in mind when analyzing the results.

Interesting cases deserving some comments can be identified on Table 1:

- 421-like cases: Those are obvious cases (421, 111, 611, ...). If the player gets those high score combinations, the policy is 000; keep the high score, that is, do not re-roll.
- 532-like cases: The opposite of 421-like cases. If you get a bad combination (522, 322, 553, ...), the policy is 111, that is, just re-roll all dices. The expected value of re-rolling all dice is better than trying to keep interesting dices.
- 221-like cases: Those are the situations where it is advisable to keep some of the dices to get interesting combinations. A good example is 211: just roll the dice 2. Then,
  - If the dice scores 3, 4, 5, 6 or even 1, the number of points increases.
  - If the dice scores 2 the number of points stay the same.
- 543-like cases: Those are similar to the previous case but with a different strategy during the first and the second reroll. Indeed the expected score is 1.61 without re-rolls and 3.10 if (one or two) re-rolls are allowed. During

the first re-roll, the expected score is therefore 3.10 and is lowered to 1.61 during the second. This is why 543 (with a score of 2) has different mixed strategies according to the number of re-rolls.

Moreover, the optimal mixed policy obtained by the soft value iteration with intermediate values of θ is reported for the previous interesting cases on Figure 3. For cases 421, 532, 221-like, the strategy is very similar during the first or second re-roll. Notice that the increase of each bin is monotonic with the increase of θ and that the function interpolates between a uniform distribution (equal to priors, when θ is small) and the quasi-deterministic (sum of Kronecker delta, when θ is high) optimal policy.

Finally, Figure 2 represents the evolution of the average number of final tokens after the game (the higher, the better, reported as mean *reward* averaged over $10e6$ runs of whole turns) and the average entropy in function of θ. The largest expected reward and minimal entropy are achieved when θ is small and the opposite is true when θ is large. The resulting functions are both logistic-shaped between two bounds:

- When θ is small, entropy is maximum as each action has a 1/8 probability to be chosen. The mean score is minimum and is the same as if we consider a random walk for this MDP. In this particular case, the expected score is the same as if no re-roll is allowed.

- When θ is large, entropy is minimum and the mixed policy converges to the optimal, deterministic, policy provided by the standard value iteration algorithm. The mean score is maximum and the corresponding policy induces re-rolls.

This example clearly shows that using a mixed strategy allows to balance the strength of the player.

6 Conclusion

This work presented a procedure for solving randomized Markov decision processes. The problem is viewed as a constrained randomized shortest path problem on a bipartite graph.

The first one is similar to the value-iteration method for solving Markov decision processes and is iterative. Its main drawback is that it is computationally demanding since it relies on iterative algorithms and necessitates the solution of two linear systems of equations at each iteration. On the other hand, it exploits the sparseness of the network and relies on an iterative scheme which can be useful when continual adaptation is needed, for instance in changing environments.

The second procedure was originally introduced by Akamatsu in the framework of transportation networks. Based on Akamatsu's ideas, and by revisiting the problem from a statistical physics perspective, we show that randomized shortest-path problems can be computed efficiently by solving a simple linear system of equations. This still is not very efficient in comparison with state-of-the-art algorithms solving single-source single-destination shortest-path problems since solving a system of linear equations is $O(n^3)$ where n is the number of unknowns. However, for some sparse graphs having a special structure, this method could eventually prove useful; this will be investigated in further works.

Further works will be devoted to the analysis of the algorithm and to the design of other procedures, differing in the definition of the global entropy quantifying the randomness in the network. We also plan to tackle Markov decision processes, as well as multiple-sources multiple-destinations problems, with this approach. Furthermore, during the time of submission of this paper, we already exploited the randomized shortest-path distance as a dissimilarity measure between nodes for nodes clustering or betweenness computation ([**?**]). A covariance measure between nodes could also be defined within the same sum-over-paths framework: two nodes would be considered as correlated if they often co-occur on the same path.

A further application would be to design randomized edit distances or kernel-based sequence alignment procedures accounting for all editing/alignment paths. Yet another application would be to extend some of the results to the so-called semiring framework ([**?**, **?**, **?**]). Finally, we also plan to investigate the links between our proposed models and the recent work of [**?**].

Acknowledgements

This work was partially supported by the Immediate and the Brufence projects funded by InnovIris (Brussels region), as well as former projects funded by the Walloon region, Belgium. We thank these institutions for giving us the opportunity to conduct both fundamental and applied research.

We also thank Benjamin Blaise, a former Master student, who helped us to investigate the randomized Markov decision processes in his Master thesis.

Appendix: Proof of the main results

A Derivation of the soft value iteration algorithm

In order to compute the optimal policy p^*_{ku}, we observe from Equation (11) that we need to find the backward variable z_{un} starting from an action $u \in \mathcal{A}$,

$$p^*_{ku} \propto p^{\text{ref}}_{ku} \exp[-\theta c_{ku}] z_{un}$$

in our bipartite graph. The quantity p^*_{ku} then needs to be normalized so that $\sum_{u \in \mathcal{U}(k)} p^*_{ku} = 1$.

A.1 Computation of the backward variable on action nodes

Now, from the definition of the backward variable (Equation (8), but including the augmented costs), we obtain by decomposing the paths $u \rightsquigarrow n$ into the first step $u \to l$, and then the remaining steps $l \rightsquigarrow n$ (see [?] for a related computation)

$$\begin{aligned}
z_{un} &= \sum_{\wp_{un} \in \mathcal{P}_{un}} \tilde{\pi}(\wp_{un}) \exp[-\theta \tilde{c}'(\wp_{un})] \\
&= \sum_{l \in \mathcal{S}ucc(u)} \sum_{\wp_{ln} \in \mathcal{P}_{ln}} p^{\text{ref}}_{ul} \tilde{\pi}(\wp_{ln}) \exp[-\theta(c'_{ul} + \tilde{c}(\wp_{ln}))] \\
&= \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \exp[-\theta c'_{ul}] \underbrace{\sum_{\wp_{ln} \in \mathcal{P}_{ln}} \tilde{\pi}(\wp_{ln}) \exp[-\theta \tilde{c}(\wp_{ln})]}_{z_{ln}} \\
&= \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \exp[-\theta c'_{ul}] z_{ln} \qquad (40)
\end{aligned}$$

where $\wp_{un}$ is a path starting in action node u and ending in the killing absorbing state n. Note that we have to use the augmented costs c'_{ul} in order to ensure that the flow in the edge (u, l) is equal to the predefined transition probability p^{ref}_{ul} provided by the environment. The value of these augmented costs can be found in Equation (33) and will be adapted to our bipartite MDP graph later. This provides

$$c'_{ul} = \tfrac{1}{\theta} \log z_{ln} - \tfrac{1}{\theta} \sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} \log z_{l'n} + \sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} c_{ul'} \qquad (41)$$

Injecting this result in Equation (40) yields

$$\begin{aligned}
z_{un} &= \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \exp[-\theta c'_{ul}] z_{ln} \\
&= \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \exp[-\log z_{ln}] \exp\Bigg[\sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} \log z_{l'n} \Bigg] \exp\Bigg[-\theta \sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} c_{ul'} \Bigg] z_{ln} \\
&= \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \exp\Bigg[\sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} \log z_{l'n} \Bigg] \exp\Bigg[-\theta \sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} c_{ul'} \Bigg] \\
&= \exp\Bigg[\sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \log z_{ln} \Bigg] \exp\Bigg[-\theta \sum_{l' \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul'} c_{ul'} \Bigg] \qquad (42)
\end{aligned}$$

Now, in the case of our bipartite MDP graph, the original costs c_{ul} are equal to zero for transitions between action nodes and state nodes, as specified in Equation (35). Therefore,

$$z_{un} = \exp\Bigg[\sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \log z_{ln} \Bigg] \qquad (43)$$

which is the expression for computing the backward variables associated to actions. However, it depends on z_{ln}.

A.2 Computation of the optimal policy

Let us now replace the value of z_{un}, just obtained in the previous section, in the equation providing the optimal policy, i.e., Equation (40),

$$\begin{aligned}
p^*_{ku} &\propto p^{\text{ref}}_{ku} \exp[-\theta c_{ku}] z_{un} \\
&= p^{\text{ref}}_{ku} \exp[-\theta c_{ku}] \exp\Bigg[\sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \log z_{ln} \Bigg] \\
&= p^{\text{ref}}_{ku} \exp\Bigg[-\theta \Big(c_{ku} + \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} (-\tfrac{1}{\theta} \log z_{ln}) \Big) \Bigg] \\
&= p^{\text{ref}}_{ku} \exp\Bigg[-\theta \Big(c_{ku} + \sum_{l \in \mathcal{S}ucc(u)} p^{\text{ref}}_{ul} \phi^{\mathcal{S}}_l(T) \Big) \Bigg] \qquad (44)
\end{aligned}$$

which justifies Equation (39) after normalization. In the last step, we introduced the free energy defined in Subsection 4.2.1, $\phi^{\mathcal{S}}_k(T) = -T \log z_{kn} = -\frac{1}{\theta} \log z_{kn}$. Because the free energy, and thus the optimal policy, depends on the backward variable defined on states, z_{kn} with $k \in \mathcal{S}$, let us now turn to the computation of this quantity.

A.3 Computation of the backward variable on states

By proceeding as for the derivation of the recurrence relation of Equation (40), we obtain

$$\begin{aligned} z_{kn} &= \sum_{\wp_{kn} \in \mathcal{P}_{kn}} \tilde{\pi}(\wp_{kn}) \exp[-\theta \tilde{c}(\wp_{kn})] \\ &= \sum_{u \in \mathcal{U}(k)} p_{ku}^{\text{ref}} \exp[-\theta c_{ku}] z_{un} \end{aligned} \tag{45}$$

and this last result only depends on the original costs c_{ku} (and not the augmented costs c'_{ku}) because there is no augmented costs associated to the transitions from a state to an action node, as these transitions are not part of the set of constrained transitions (see Equation (35)).

A.4 The soft value iteration recurrence equation

Taking $-\frac{1}{\theta}\log$ of each side of this equation (45), using $\phi_k^{\mathcal{S}}(T) = -\frac{1}{\theta}\log z_{kn}$, and replacing z_{un} by its value provided in Equation (43) gives

$$\begin{aligned} \phi_k^{\mathcal{S}}(T) &= -\tfrac{1}{\theta}\log\Bigg[\sum_{u\in\mathcal{U}(k)} p_{ku}^{\text{ref}} \exp[-\theta c_{ku}] z_{un}\Bigg] \\ &= -\tfrac{1}{\theta}\log\Bigg[\sum_{u\in\mathcal{U}(k)} p_{ku}^{\text{ref}} \exp[-\theta c_{ku}]\ \exp\Big[\sum_{l\in\mathcal{S}ucc(u)} p_{ul}^{\text{ref}} \log z_{ln}\Big]\Bigg] \\ &= -\tfrac{1}{\theta}\log\Bigg[\sum_{u\in\mathcal{U}(k)} p_{ku}^{\text{ref}} \exp\Big[-\theta\Big(c_{ku} + \sum_{l\in\mathcal{S}ucc(u)} p_{ul}^{\text{ref}}(-\tfrac{1}{\theta}\log z_{ln})\Big)\Big]\Bigg] \\ &= -\tfrac{1}{\theta}\log\Bigg[\sum_{u\in\mathcal{U}(k)} p_{ku}^{\text{ref}} \exp\Big[-\theta\Big(c_{ku} + \sum_{l\in\mathcal{S}ucc(u)} p_{ul}^{\text{ref}}\phi_l^{\mathcal{S}}(T)\Big)\Big]\Bigg] \end{aligned} \tag{46}$$

which is exactly the sought result of Equation (37), together with the fact that when $k = n$ (absorbing, killing, state), $z_{nn} = 1$ (see Equation (8)) and thus $\phi_n^{\mathcal{S}}(T) = 0$. Notice that the $\phi_k^{\mathcal{S}}(T) = 0$ are necessarily non-negative. This equation is a smooth approximation of the standard value iteration algorithm through the softmin operator (Equation (17)) – the soft value iteration – that has to be iterated on all nodes until convergence.

B Convergence of the soft value iteration algorithm

As shown in [**?**] in the context of distributed consensus where a similar equation appears, the iterative update of Equation (46) (the soft version of the Bellman-Ford algorithm) converges, and is independent of the initial values. We will now prove the same result for the soft value iteration algorithm, but with a different argument based on the fixed point theorem and contraction mapping.

First, let us observe that the solution to the recurrence relation of the Bellman-Ford equation (37) (first line of the equation) is invariant up to a translation of the origin. Indeed, it can easily be shown that if $\phi_k^{\mathcal{S}}(T)$ is a solution of (37), a shift of the free energy by a quantity α, that is ${\phi'}_k^{\mathcal{S}}(T) = \phi_l^{\mathcal{S}}(T) + \alpha$, is also a solution to (37). To overcome this underdetermination, the free energy is set to zero on the absorbing, goal, node n, $\phi_n^{\mathcal{S}}(T) = 0$. We will therefore study the following fixed point iteration restricted to states $\{1, 2, \ldots, n-1\}$ only,

$$\phi_k^{\mathcal{S}}(T) \leftarrow -\tfrac{1}{\theta} \log \left[\sum_{u \in \mathcal{U}(k)} p_{ku}^{\text{ref}} \exp \Big[-\theta \Big(c_{ku} + \sum_{l=1}^{n-1} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T) \Big) \Big] \right] \text{ for } k = 1, \ldots, n-1 \tag{47}$$

Then, it is well-known that this fixed-point iteration converges to a unique solution in a convex domain if the Jacobian matrix, $\mathbf{J}$, of the transformation has a matrix norm (for instance its spectral radius) smaller than 1 in this domain [**?**, **?**, **?**]. In that case, the fixed-point transformation is what is called a **contraction mapping**. We will thus compute the spectral radius of the Jacobian matrix and verify that it is smaller than one for all non-negative values of $\phi_k^{\mathcal{S}}(T)$.

The element i, j of this Jacobian matrix can easily be computed from Equation (47),

$$\frac{\partial \phi_i^{\mathcal{S}}}{\partial \phi_j^{\mathcal{S}}} = \frac{\displaystyle\sum_{u \in \mathcal{U}(i)} p_{iu}^{\text{ref}} \exp \Big[-\theta \Big(c_{iu} + \sum_{l=1}^{n-1} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T) \Big) \Big] p_{uj}^{\text{ref}}}{\displaystyle\sum_{u' \in \mathcal{U}(i)} p_{iu'}^{\text{ref}} \exp \Big[-\theta \Big(c_{iu'} + \sum_{l'=1}^{n-1} p_{u'l'}^{\text{ref}} \phi_{l'}^{\mathcal{S}}(T) \Big) \Big]} \quad \text{for } i, j = 1, \ldots, n-1 \tag{48}$$

Then, we can verify that this matrix is sub-stochastic as row sums are always lesser or equal to 1 and some row sums are strictly less that 1 (those for which $p_{un}^{\text{ref}} > 0$) for all positive $\theta > 0$. Indeed, defining

$$\gamma_{iu} = \frac{p_{iu}^{\text{ref}} \exp \Big[-\theta \Big(c_{iu} + \sum_{l=1}^{n-1} p_{ul}^{\text{ref}} \phi_l^{\mathcal{S}}(T) \Big) \Big]}{\displaystyle\sum_{u' \in \mathcal{U}(i)} p_{iu'}^{\text{ref}} \exp \Big[-\theta \Big(c_{iu'} + \sum_{l'=1}^{n-1} p_{u'l'}^{\text{ref}} \phi_{l'}^{\mathcal{S}}(T) \Big) \Big]} \leq 1 \tag{49}$$

so that $\sum_{u \in \mathcal{U}(i)} \gamma_{iu} = 1$, the elements of the Jacobian matrix $\mathbf{J}$ can be rewritten as

$$[\mathbf{J}]_{ij} = \frac{\partial \phi_i^{\mathcal{S}}}{\partial \phi_j^{\mathcal{S}}} = \sum_{u \in \mathcal{U}(i)} \gamma_{iu} p_{uj}^{\text{ref}} \tag{50}$$

Because $\sum_{j=1}^{n} p_{uj}^{\text{ref}} = 1$, $\sum_{j=1}^{n-1} p_{uj}^{\text{ref}} \leq 1$ for each action u holds and the inequality is strict when $p_{un}^{\text{ref}} > 0$. Let us compute the row sums of this Jacobian matrix,

$$\begin{aligned}
\sum_{j=1}^{n-1} \frac{\partial \phi_i^{\mathcal{S}}}{\partial \phi_j^{\mathcal{S}}} &= \sum_{j=1}^{n-1} \sum_{u \in \mathcal{U}(i)} \gamma_{iu} p_{uj}^{\text{ref}} \\
&\leq \sum_{j=1}^{n} \sum_{u \in \mathcal{U}(i)} \gamma_{iu} p_{uj}^{\text{ref}} \\
&= \sum_{u \in \mathcal{U}(i)} \gamma_{iu} = 1
\end{aligned} \tag{51}$$

and the inequality is strict for rows with $p_{un}^{\text{ref}} > 0$. Consequently, because, in addition, all the elements of the matrix are non-negative, $\mathbf{J}$ is sub-stochastic [**?**].

Finally, from the definition of the Jacobian matrix (48), the graph induced by $\mathbf{J}$ connects the $n-1$ states of the bipartite graph in the following way: node i and node j are connected if and only if it is possible to reach node j in two steps (there exists at least one action u such that $(p_{iu}^{\text{ref}} * p_{uj}^{\text{ref}})$ is different from zero) in the bipartite graph. In other words, we recover the connectivity between states of the bipartite graph. As it is assumed that the absorbing, killing, node n can be reached from any initial node of the graph, this property is kept in $\mathbf{J}$, meaning that, exactly as for a standard absorbing Markov chain, the total probability mass in the transient states of the network (nodes 1 to $n-1$) will gradually decrease and $\lim_{t\to\infty} \mathbf{J}^t \to 0$. This implies that the spectral radius of the Jacobian matix $\mathbf{J}$ is strictly less than 1. Therefore, as the spectral radius is a matrix norm, the soft value iteration converges to a unique solution independently of the initial conditions [**?**, **?**, **?**].

References

[1] Y. Achbany, F. Fouss, L. Yen, A. Pirotte, and M. Saerens. Tuning continual exploration in reinforcement learning: an optimality property of the boltzmann strategy. *Neurocomputing*, 71:2507–2520, 2008.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: Theory, algorithms, and applications.* Prentice Hall, 1993.

[3] T. Akamatsu. Cyclic flows, markov process and stochastic traffic assignment. *Transportation Research B*, 30(5):369–386, 1996.

[4] K. Arrow, L. Hurwicz, and H. Uzawa. *Studies in linear and non-linear programming.* Stanford University Press, 1958.

[5] D. P. Bertsekas. *Network optimization: continuous and discrete models.* Athena Scientific, 1998.

[6] D. P. Bertsekas. *Nonlinear Programming, 2nd ed.* Athena Scientific, 1999.

[7] D. P. Bertsekas. *Dynamic programming and optimal control.* Athena Scientific, 2000.

[8] D. P. Bertsekas and J. Tsitsiklis. *Neuro-dynamic programming.* Athena Scientific, 1996.

[9] A. Bjorck and G. Dahlquist. *Numerical methods in scientific computing, volume II.* SIAM, unpublished working copy edition, 2008.

[10] B. Blaise. Randomized markov decision processes: a study of two new algorithms. Master's thesis, Universite de Louvain, 2013. Promotor: Prof. Marco Saerens.

[11] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge University Press, 2004.

[12] B. Carre. *Graphs and networks.* Oxford University Press, 1979.

[13] N. Christofides. *Graph theory: An algorithmic approach.* Academic Press, 1975.

[14] J. Cook. Basic properties of the soft maximum. Unpublished manuscript available from www.johndcook.com/blog/2010/01/13/soft-maximum, 2011.

[15] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to algorithms, 3th Edition.* The MIT Press, 2009.

[16] T. M. Cover and J. A. Thomas. *Elements of information theory, 2nd ed.* John Wiley and Sons, 2006.

[17] J. Culioli. *Introduction a l'optimisation.* Ellipses, 2012.

[18] G. Dahlquist and A. Bjorck. *Numerical methods.* Prentice-Hall, 1974.

[19] F. Fouss, M. Saerens, and M. Shimbo. *Algorithms and models for network data and link analysis.* Cambridge University Press, 2016.

[20] S. García-Díez, F. Fouss, M. Shimbo, and M. Saerens. A sum-over-paths extension of edit distances accounting for all sequence alignments. *Pattern Recognition*, 44(6):1172–1182, 2011.

[21] M. Gondran and M. Minoux. *Graphs and algorithms.* John Wiley & Sons, 1984.

[22] I. Griva, S. Nash, and A. Sofer. *Linear and nonlinear optimization.* SIAM, 2nd edition, 2008.

[23] E. T. Jaynes. Information theory and statistical mechanics. *Physical review*, 106:620–630, 1957.

[24] D. Jungnickel. *Graphs, networks, and algorithms, 3th ed.* Springer, 2008.

[25] J. N. Kapur and H. K. Kesavan. *The generalized maximum entropy principle (with applications).* Sandford Educational Press, 1987.

[26] J. G. Kemeny and J. L. Snell. *Finite Markov chains.* Springer-Verlag, 1976.

[27] I. Kivimäki, B. Lebichot, J. Saramäki, and M. Saerens. Two betweenness centrality measures based on randomized shortest paths. *Scientific Reports*, 6:srep19668, 2016.

[28] I. Kivimäki, M. Shimbo, and M. Saerens. Developments in the theory of randomized shortest paths with a comparison of graph node distances. *Physica A: Statistical Mechanics and its Applications*, 393:600–616, 2014.

[29] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML-94)*, pages 157–163, 1994.

[30] C. D. Meyer. *Matrix analysis and applied linear algebra.* SIAM, 2000.

[31] M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. *Journal of automata, languages and combinatorics*, 7(3):321–350, 2002.

[32] R. Motwani and P. Raghavan. *Randomized algorithms.* Cambridge University Press, 1995.

[33] Y. Nesterov. Characteristic functions of directed graphs and applications to stochastic equilibrium problems. *Optimization and Engineering*, 8(2):193–214, 2007.

[34] J. Norris. *Markov chains.* Cambridge University Press, 1997.

[35] M. J. Osborne. *An introduction to game theory.* Oxford University Press, 2004.

[36] L. Peliti. *Statistical mechanics in a nutshell.* Princeton University Press, 2011.

[37] G. Phillips and P. Taylor. *Theory and applications of numerical analysis, 2nd ed.* Academic Press, 1996.

[38] W. Powell. *Approximate dynamic programming, 2nd ed.* John Wiley and Sons, 2011.

[39] M. Puterman. *Markov decision processes: discrete stochastic programming.* John Wiley and Sons, 1994.

[40] H. Raiffa. *Decision analysis.* Addison-Wesley, 1970.

[41] R. Rardin. *Optimization in operations research.* Prentice Hall, 1998.

[42] L. Reichl. *A modern course in statistical physics, 2nd ed.* Wiley, 1998.

[43] M. Saerens, Y. Achbany, F. Fouss, and L. Yen. Randomized shortest-path problems: Two related models. *Neural Computation*, 21(8):2363–2404, 2009.

[44] R. Sedgewick. *Algorithms, 4th ed.* Addison-Wesley, 2011.

[45] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction.* The MIT Press, 1998.

[46] A. Tahbaz and A. Jadbabaie. A one-parameter family of distributed consensus algorithms with boundary: from shortest paths to mean hitting times. In *Proceedings of IEEE Conference on Decision and Control*, pages 4664–4669, 2006.

[47] H. C. Tijms. *A first course in Stochastic Models.* John Wiley and Sons, 2003.

[48] E. Todorov. Linearly-solvable markov decision problems. In *Advances in Neural Information Processing Systems 19 (NIPS 2006)*, pages 1369–1375. MIT Press, 2006.

[49] D. White. Real applications of Markov decision processes. *Interfaces*, 15(6):73–83, 1985.

[50] D. White. Further real applications of Markov decision processes. *Interfaces*, 18(5):55–61, 1988.

[51] D. J. White. A survey of applications of Markov decision processes. *Journal of the Operational Research Society*, 44(11):1073–1096, 1993.

[52] L. Yen, A. Mantrach, M. Shimbo, and M. Saerens. A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances. In *Proceedings of the 14th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008)*, pages 785–793, 2008.