Solving Ramified Optimal Transport Problem in the Bayesian Influence Diagram Framework

Michal Matuszak[1], Jacek Miękisz[2], and Tomasz Schreiber*[1]

[1] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Chopina 12/18, 87–100 Torun, Poland
{gruby, tomeks}@mat.umk.pl
[2] Institute of Applied Mathematics and Mechanics, University of Warsaw, Banacha 2, 02–097 Warsaw, Poland
miekisz@mimuw.edu.pl

Abstract. The goal of the ramified optimal transport is to find an optimal transport path between two given probability measures. One measure can be identified with a source while the other one with a target. The problem is well known to be NP–hard. We develop an algorithm for solving a ramified optimal transport problem within the framework of Bayesian networks. It is based on the decision strategy optimisation technique that utilises self–annealing ideas of Chen–style stochastic optimisation. Resulting transport paths are represented in the form of tree–shaped structures. The effectiveness of the algorithm has been tested on computer–generated examples.

Keywords: Optimal Transport Path, Transport Network, Branching Structure, Bayesian Influence Diagrams, Optimal Decision Strategies

1 Introduction

The transport problem was introduced by G. Monge in a very famous paper, *Mémoire sur la théorie des déblais et des remblais* [10,4]. Recently this classical problem has gained an extensive popularity [1,14]. The original problem is to move a pile of soil from one place to another with the minimal effort. In 1942, Kantorovich introduced his formalization of a relaxed version of the Monge's problem [7,2]. The task of finding optimal paths was transformed into the problem of transporting a positive measure μ_s onto another positive measure μ_d with the same mass. The Monge–Kantorovich approach assumes that the transport cost is proportional to the distance and the transported mass. It favours thin routes rather than wide ones. Unfortunately it is not practical from the economic point of view.

In most transport networks, sending each particle straight to the destination is economically unrealistic. The preferable solution is to aggregate particles and move them together as it happens in tree leafs or on highways. We should mention

* Deceased author (1975 – 2010).

the Steiner tree problem where one minimizes only the total length of a network [13] and omits the cost of constructing edges. His model is not appropriate for our purposes because it does not discriminate the cost of high or low capacity edges; constructing a high–capacity highway is more expensive than constructing a backroad.

The first model taking into account the cost of edges was introduced by Gilbert and it has been extensively investigated in [14]. He showed that in shipping two objects from nearby cities to the same far away city, see Fig. 1(c), it may be more optimal to first transport them to a common location and then transport them together to the target. In this case, a Y shaped path is preferable to a V shaped path, see Fig. 1. In general, resulted paths form leaf–like structures. Biological leafs tend to maximize an internal efficiency by developing an efficient transport system for water and nutrients [16]. We should note that the presented problems are NP–hard [4,13].

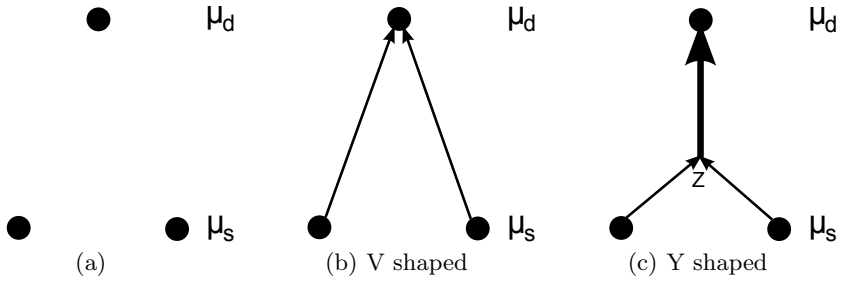


Fig. 1. (a) Three cities, two of them at the bottom are the source and the third city at the top is the destination for transported goods, (b) Monge–Kantorovich solution, (c) Gilbert solution, the interior vertex z can be determined analytically [14].

An influence diagram [6,11,12] is an extension of a Bayesian network [8,6,11,12], in which not only a probabilistic inference occurs but also decision making problems are solved. Influence diagrams are built on a directed acyclic graphs (DAGs) whose nodes and edges have standard interpretations stemming from and extending those used for Bayesian networks.

An influence diagram, similar to a Bayesian network, can be built with the use of chance nodes, which we represent as ovals. Also two additional types of nodes are introduced: decision nodes corresponding to available decisions (rectangles) and utility nodes (rhombi) specifying payoff functions (utilities) to be maximized by suitable choices of decision policies.

If the network is well designed, then the arcs leading to chance nodes specify direct causal relationships not necessarily corresponding to any temporal ordering. The arcs leading to decision nodes indicate the information available at the moment of decision making, thus feeding input to decision policies. The influence

diagrams can be considered as generalizations of (symmetric) decision trees, see [6].

Finding an optimal decision strategy for an influence diagram is an NP–hard task. One can show this easily by reducing the traveling salesman problem (that is NP–complete) to our task.

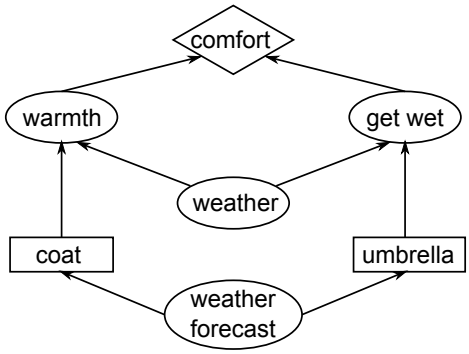


Fig. 2. Illustration of a simple influence diagram that includes two decision nodes: whether or not to take an umbrella and/or a coat for a journey. The decisions have an impact on the warmth and can prevent from getting wet from the rain, but if it does not rain, then carrying an umbrella has a negative impact on the mood.

In Fig. 2, an example of an influence diagram is shown. The decision node *umbrella* represents the choice whether or not to take an umbrella. Taking an umbrella does increase the chance of not getting wet during rain, yet it also causes negative effects, such as the need of carrying an additional weight. Further, wearing a *coat* decreases the chance of getting chilled but also negatively affects our comfort if the outside temperature is too high. All these effects are jointly taken into account in the utility node *comfort*.

There exists a number of algorithms for finding an optimal decision strategy for an influence diagram. For a detailed discussion we refer the reader to Chapter 10 in [6] and Subsection 5.2.2 in [11]. We focus our attention on the Chen–style [5] stochastic optimisation algorithm described in [9] which is well suited for our task.

In Section 2, we formalize the transport problem. Then we outline the transformation of the problem into influence diagrams. Results, technical details, and a discussion are contained in following sections.

2 Optimal transport problem

In this section we recall some concepts of Xia [14,15,16] concerning optimal transport paths between measures.

Let (X, d) be a metric space. We define an atomic measure on X as follows

$$a = \sum_{i=1}^{k} a_i \delta_{x_i} \tag{1}$$

for some integer k and points $x_i \in X$, a_i are positive numbers and δ_{x_i} is the Dirac mass located at the point x_i. We will work with the probability measures, i.e. we assume that $\sum_{i=1}^{k} a_i = 1$.

Let $A(X)$ be the space of all atomic probability measures on X. For measures on X,

$$\mu_s = \sum_{i=1}^{k} s_i \delta_{x_i} \text{ and } \mu_d = \sum_{j=1}^{n} d_j \delta_{y_j} \tag{2}$$

a **transport path** from μ_s (source) to μ_d (destination) is defined as a weighted directed acyclic graph (DAG) $G = (V_G, E_G)$, where V_G is a set of vertices such that $\{x_1, x_2, \ldots, x_k\} \cup \{y_1, y_2, \ldots, y_n\} \subset V_G$ and E_G is a set of directed edges with a weight function

$$w : E_G \to (0, +\infty). \tag{3}$$

Hence V_G consist of source, destination and intermediate vertices, see for example Fig. 1(c) and Fig. 3. The value $w(e)$ can be identified with the amount of goods transported along the edge e.

The balance equation for every $v \in V_G$

$$\sum_{e \in E_G, e^- = v} w(e) = \sum_{e \in E_G, e^+ = v} w(e) + \begin{cases} s_i \text{ , if } v = x_i \text{ for some } i = 1, \ldots, k \\ -d_j \text{ , if } v = y_j \text{ for some } j = 1, \ldots, n \\ 0 \text{ , otherwise} \end{cases} \tag{4}$$

where e^- denotes the first vertex of the edge $e \in E_G$ and e^+ is the second vertex. It simply means that the total mass flowing into $v \in V_G$ equals to the total mass flowing out of v.

For any $0 \leq \alpha \leq 1$ and any transport path G, we define the **path cost** function $w_p(G, \alpha)$ as follows

$$w_p(G, \alpha) = \sum_{e \in E_G} \|e\| * [w(e)]^{\alpha} \tag{5}$$

where $\|e\|$ denotes the length of the edge e.

The ramified optimal transport problem focuses on finding a transport path from μ_s to μ_d which minimizes $w_p(G, \alpha)$. The minimizer is called an optimal transport path. In other words, for a given G, and α we have to create a weight function such that (4) is satisfied and (5) is minimized.

3 The algorithm

In this section, the formal description of our algorithm is given. First, we define the total cost function which is minimized during the optimization phase. Then

we present the transformation of the optimal transport problem into an influence diagram and finally we translate an optimized decision policy into an optimal transport path.

So far we have assumed that each destination node receives a specific amount of mass. Such a strict constraint prevents us from applying many optimization techniques so we relax the above assumption and introduce a **disagreement cost** for a DAG G,

$$w_d(G) = \sum_{j=1}^{n} \left[d_j - \sum_{e \in E_G, e^+ = v} w(e) \right]^2 \tag{6}$$

which characterizes the difference between a shipped and expected mass.

We define the **total cost function** which is based on (5) and (6),

$$w(G, \alpha, c_1, c_2) = c_1 w_p(G, \alpha) + c_2 w_d(G), \tag{7}$$

where c_1 and c_2 are weights which control the importance of the **disagreement cost** and the **path cost**. The objective is to minimize the total cost function in (7) which is identified with the $-$payoff ("minus" because it is assumed that we maximize the payoff function) in the influence diagram.

Let us assume that an influence diagram $(\mathcal{S}, \mathcal{P}, \mathcal{U})$ is given, which is built on a connected DAG $\mathcal{S}$, with conditional probability tables (CPTs) $\mathcal{P}$ and utility functions $\mathcal{U}$. The set of nodes in $\mathcal{S}$ splits into chance nodes $C_{\mathcal{S}}$, decision nodes $D_{\mathcal{S}}$ and utility nodes $U_{\mathcal{S}}$. An influence diagram that describes an optimal transport problem is constructed in the following way:

- $C_{\mathcal{S}} = \emptyset$
- $D_{\mathcal{S}} = V_G \backslash \{y_j \| j = 1, \ldots, n\}$
- $U_{\mathcal{S}} = \{y_j \| j = 1, \ldots, n\}$

In addition, for each decision node $D \in D_{\mathcal{S}}$, a *randomised policy* τ_D is attached. It assigns to each configuration $\bar{w}$ of $\mathrm{pa}(D)$ (where $\mathrm{pa}(D)$ is a set of parents of node D) a probability distribution on possible decisions to be taken, that is to say $\tau_D(d|\bar{w})$ stands for the probability of choosing a decision d given that $\mathrm{pa}(D) = \bar{w}$. These randomised policies will evolve in the course of the optimisation process, eventually to become (sub)optimal deterministic policies which collectively determine the utility maximizing strategy for the influence diagram considered. The initial choice of τ_D, $D \in D_{\mathcal{S}}$ can be either *uniform*, with all routes equiprobable, or *heuristic*, provided some additional knowledge is available allowing us to make a good *first guess* about the optimal path.

The connections in $\mathcal{S}$ are replicated from the set of edges E_G. If V_G consists only of source and destination vertices and does not have intermediate ones, then we can add them either

- uniformly – producing a regular grid of vertices
- heuristically – an additional knowledge about preferred paths is provided
- randomly – all parts of the space are treated with equal probabilities

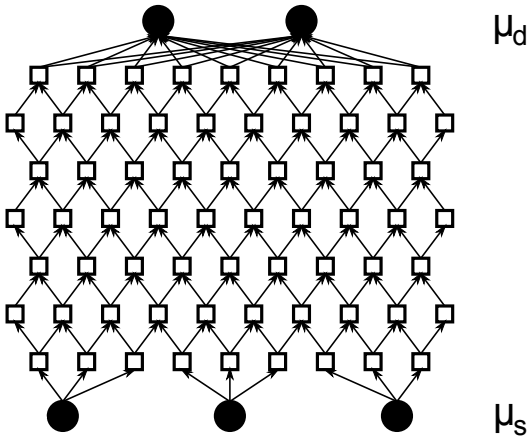


Fig. 3. Representation of an influence diagram used in the algorithm. Squares describe intermediate vertices where junctions can occur and dots represent source and destination of the mass.

Next we have to define connections between them. To preserve an acyclic property and equality of routes we assume that the atomic measures (the source and the destination) can be spatially separated by a hyperplane. Edges can by defined as follows,

1. For each decision node we define a maximal number of children k_d
2. $Q = \{x_i \| i = 1, \ldots, k\}$ and $R = \emptyset$
3. For each $q \in Q$, find k_q nearest neighbours of q, set them as children of q, and add to R.
4. If $R \neq \emptyset$ then $Q = R$ and $R = \emptyset$ and go to 3.

For such an influence diagram it is feasible to use the stochastic optimisation algorithm from [9]. Description of the algorithm falls beyond the scope of the present article. Results of the algorithm will be stored in the *randomised policy* τ_D. Using computed policies we can easily determine the optimal paths. Each policy describes where and how we should transport the incoming mass. Starting from roots of DAG we transport the source mass to the children according to computed policies.

4 Examples

In the first example, presented in Fig. 4, we reproduced the Gilbert solution from Fig. 1(c). The angle between merging edges was computed in [14] and is equal to $arccos(2^{2\alpha-1} - 1)$. Expected solution for $\alpha = 0.7$ is 71.36 degrees and the experimental value obtained from presented algorithm is equal to 73.5 degrees and highly depends on the distribution of decision nodes. The second example is presented in Fig. 5. Simulation of the influence diagram from Fig. 5 required 160

ms time per epoch of the algorithm [9]. The resulted transport path follows the expectations and results from [16]. It favours high capacity roads over narrow ones.

The programme has been implemented in language `D` [3], currently gaining popularity as a natural successor of `C++`. The implementation, aimed so far mainly at algorithm evaluation purposes, can be described as careful but not fully performance–optimised, with the total utility evaluation performed using the standard Monte–Carlo rather than a more refined and effective scheme. All tests were performed on a machine with Intel Core 2 Q9300 2.50 GHz CPU and 4GB RAM.

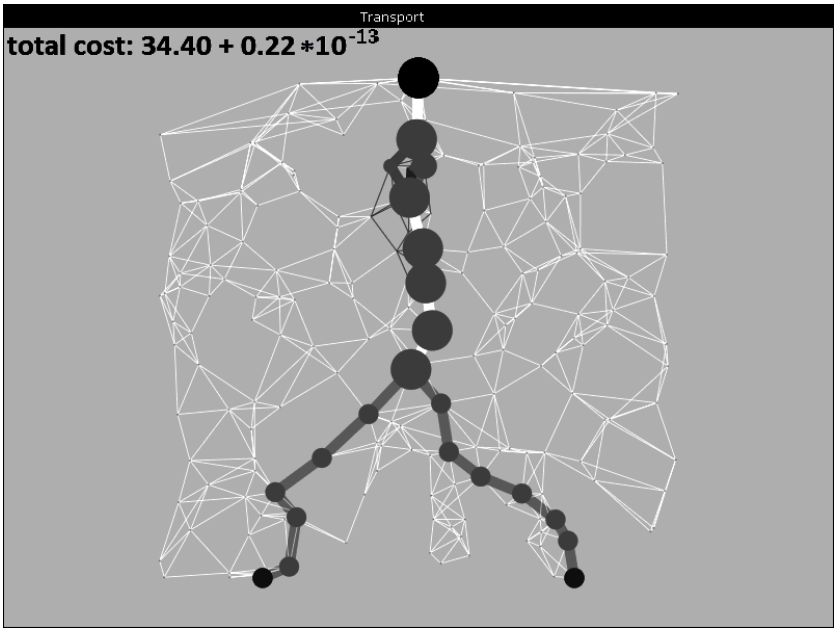
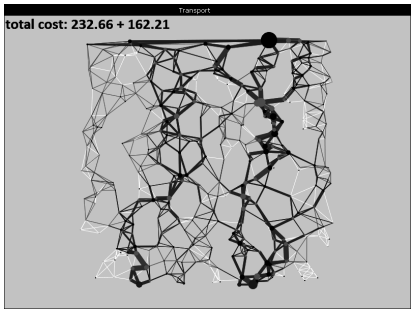


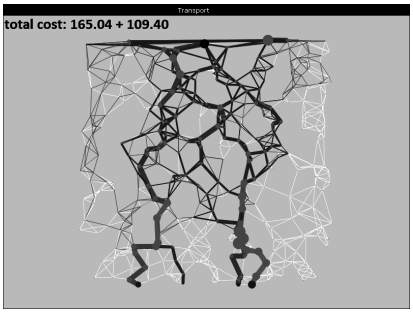
Fig. 4. Results of the algorithm on a graph that has 2 source nodes, one destination node and 200 randomly distributed decision nodes. Resulted transport path follows the Gilbert solution, see Fig. 1(c). In the upper left corner the total cost is presented in the form given by Eq. 7. Parameter α was set to 0.7, $s_1 = s_2 = 0.5$ and $d_1 = 1$.

5 Conclusions

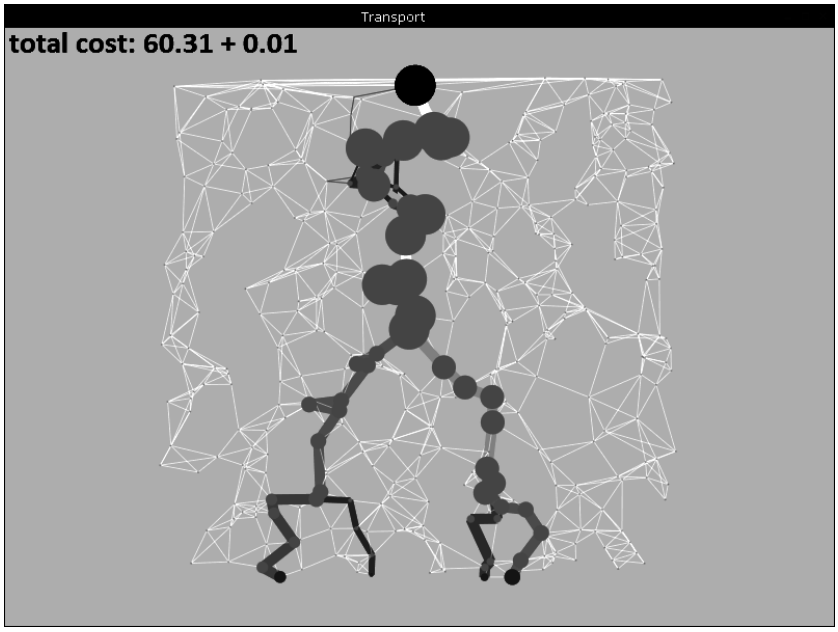
A new stochastic algorithm for solving transportation problem has been presented. The main advantage of the introduced method is its innovative application of Bayesian influence diagrams. Experimental results indicated the correctness of the algorithm. In the first test, the analytical result has been reproduced



(a) Just after start



(b) After 2000 epochs



(c) After 5000 epochs

Fig. 5. Results of the algorithm on a graph that has 4 source nodes, one destination node and 400 randomly distributed decision nodes. In the upper left corner the total cost is presented in the form given by Eq. 7. Parameter α was set to 0.7, $s_1 = 0.2$, $s_2 = 0.4$, $s_3 = 0.3$, $s_4 = 0.1$ and $d_1 = 1$.

and in the second one our expectation for the solution has also been met. The Chen's style algorithm for solving Bayesian influence diagram has been shown as a powerful tool able to find other applications in machine learning related problems.

Acknowledgements

This research has been supported by the National Science Centre grant 2011/01/N/ST6/00573 (2011-2014). The authors gratefully acknowledge the access to the PL–Grid[3] infrastructure that is co–funded by the European Regional Development Fund as a part of the Innovative Economy program. The work of M. Matuszak has also been supported by the European Social Fund as a part of the Sub–measure 4.1.1 (National PhD Programme in Mathematical Sciences).

References

1. Ambrosio, A. Lecture Notes on Optimal Transport Problems, Scuola Normale Superiore, Pisa (2000).
2. Ambrosio, A. Optimal transport maps in Monge–Kantorovich problem, *Proceedings of the ICM, Beijing* 3: 131–140 (2002).
3. Bell, K., Igesund, L.I., Kelly, S. Parker, M. Learn to Tango with D, Apress (2008).
4. Bernot M., Caselles V., Morel J.-M. Optimal Transportation Networks, *Lecture Notes in Mathematics 1955* (2009).
5. Chen, K. Simple learning algorithm for the traveling salesman problem,*Phys. Rev. E* 55: 7809–7812 (1997).
6. Jensen, F.V., Nielsen, T.D. Bayesian Networks and Decision Graphs, 2nd Ed., *Springer* (2007).
7. Kantorovich, L.V. On the transfer of masses, *Dokl. Akad. Nauk. SSSR* 37: 227–229 (1942).
8. Koski, T., Noble, J. Bayesian Networks: An Introduction, *John Wiley & Sons, Ltd* (2009).
9. Matuszak, M., Schreiber, T. A new stochastic algorithm for strategy optimisation in Bayesian influence diagrams, LNAI 6114: 574–581 (2010).
10. Monge, G. Mémoire sur la théorie des déblais et des remblais, Histoire de l'Académie Royale des Sciences de Paris, 666–704 (1781).
11. Neapolitan, R. E. Learning Bayesian Networks, *Prentice Hall Series in Artificial Intelligence*, *Pearson Prentice Hall* (2004).
12. Pearl, J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, *Morgan Kaufmann Publishers Inc.* (1988).
13. Vazirani, V. V. Approximation Algorithms *Springer–Verlag*, Berlin (2001).
14. Xia, Q. Optimal paths related to transport problems, *Communications in Contemporary Mathematics* 5: 251–279 (2003).
15. Xia, Q. Ramified optimal transportation in geodesic metric spaces, *Adv. Calc. Var.*4: 277–307 (2011).
16. Xia, Q. The formation of a tree leaf, *ESAIM. COCV* 13: 359–377 (2007).

[3] `http://www.plgrid.pl`