CS 886: Project

Matching for Second Hand Market with Application for Board Games Trading

by

Alexandre Daigle

CS 886: Multiagent Systems
David R. Cheriton School of Computer Science
University of Waterloo

due April 16, 2015

Waterloo, Ontario, Canada, 2015

Abstract

Second-Hand market – or barter exchange – is a huge trading market where items can either be sold for a negotiated amount of money or can be exchanged with another object. This kind of market allows for cycles of exchange such that everybody in the cycle gives an item to the next and received an item he wanted in return without having to find an exact matching pair. It is already known how to handle such market optimally in a one-time exchange, yet the online counterpart is an interesting variant that can provide both theoretical and practical advantages. We explore this notion with the example of board game trading through the website boardgamegeek.com.

1 Introduction

The idea of this project comes from the website boardgamegeek.com, which is the main resource site for the board gaming community. The website acts as a database containing the most comprehensive list of games, reviews, articles, ratings, session reports and forum discussions for board game aficionados. The world of board games is much richer than the classic American games such as *Monopoly* and *Life*. It includes thousands of games, most of them have sprouted from the enthusiasm of games in Germany due to their modern approach to gaming. Those modern games includes titles such as *Settlers of Catan*, *Ticket to Ride* and *Carcassonne* to name a few. With so many titles, a hobby that involves both playing and collecting games, and the addition of limited production constraints, the desire for a second-hand market has grown into an important part of the community.

`Boardgamegeek` has come to host a trading process called *math trades* which is a barter exchange for games. The idea is that each participant produces a list of games that they are willing to trade, and for each of these, produces a list of games that they are interested in receiving in return. The website then tries to match the participants such that the most games are traded during the exchange and that the preferences of the owners are respected.

2 Mathematical Model

We extend the model stated above in mathematical notation to represent more general barter exchange. A set of *items* $I = \{a_1, a_2, \ldots, a_n\}$ are to be traded amongst the participants. The owner of item a_k produce a list of items, W_k, he is willing to trade his item for. Note that these items may not be in the set of available item. The problem now consist of creating cycles $(a_{\pi_1}, a_{\pi_2}, \ldots, a_{\pi_k})$ where item a_{π_i} is given to the owner of item $a_{\pi_{i+1}}$ and where $a_{\pi_i} \in W_{i+1}$ — there is also a trade of a_{π_k} and a_{π_1} to complete the cycle. The goal is to create the permutation, π, of cycles that maximizes the number of items traded. We consider the 1-cycles of the permutation as an item that is not traded during the exchange.

We can conceive multiple scenarios in which this kind of exchanges takes place but we focus on two in particular:

- **One-Off Offline Exchange.** In this scenario, we are given all the information and the problem is to compute the optimal permutation of items amongst the participant.
- **Iterated Online Exchange.** The trading is iterated for multiple instances and at each instance some kind of trading occur. The goal is to maximize the number of items exchanged whilst still limiting the amount of time an item is in the exchange. The incoming items in future iterations are not known in advance.

3 One-Off Offline Exchange

A one-time offline exchange is a simplistic approach to any elaborate barter exchange mechanism but it is necessary since it provides well studied results that we reuse for online exchange. Looking back at the literature, we describe a reduction to the known problem of *maximum matching on a weighted bipartite graph*.

Theorem 1 *Second-hand item matching can be converted to a maximum matching problem on a weighted bipartite graph.*

Proof We construct the graph $G = (V, E)$ in the following way: we create two nodes, a_i^{G} (*give*) and a_i^{R} (*receive*), for each item a_i in the trade then we create an edge from a_i^{G} to a_j^{R} whenever $a_i \in W_j$. Finally, we add *self-edges* on each $(a_i^{\mathrm{G}}, a_i^{\mathrm{R}})$ for all i.

We define the weights so that every value is non-negative and we let the weight of the self-edges to be greater than $\sum_{i=0}^{n} \max_{(a_i,a_j)\in E\setminus\{(a_i,a_i)\}} w_{(a_i,a_j)}$.

We observe that the graph is bipartite since each G-nodes can only share edges with R-nodes and vice-versa. The graph also allows a perfect matching since the self-edges are always a valid choice. We can think of a minimum matching $\left\{(a_{i_1}^{\mathrm{G}}, a_{i_2}^{\mathrm{R}}), (a_{i_2}^{\mathrm{G}}, a_{i_3}^{\mathrm{R}}), \ldots, (a_{i_k}^{\mathrm{G}}, a_{i_1}^{\mathrm{R}})\right\}$ in this graph as being a cycle in which items are exchanged as follows $(a_{i_1}, a_{i_2}, \ldots, a_{i_k})$. Finally, since the weight of a self-edge is bigger than the weight of any cycles, we get that the minimum matching problem includes a self-edge in the matching if and only if there is not any other trade possible with the item without causing another self-edge to occur. ∎

Remark The minimum matching problem can be converted to a maximum matching problem by the following transformation:

$$w'_{(a^{\mathrm{G}}_{i_1}, a^{\mathrm{R}}_{i_2})} := \max_{(a^{\mathrm{G}}_{j_1}, a^{\mathrm{R}}_{j_2}) \in E} w_{(a^{\mathrm{G}}_{j_1}, a^{\mathrm{R}}_{j_2})} - w_{(a^{\mathrm{G}}_{i_1}, a^{\mathrm{R}}_{i_2})}.$$

The following known result gives the solution to the offline problem.

Theorem 2 *The maximum matching problem on a weighted bipartite graph is in $\mathcal{P}$.*

Proof The problem can be solved using what is known as the *Hungarian Method* (also known as the *Kuhn–Munkres algorithm*). The algorithm runs in $O(n^3)$ where n is the number of vertex in the graph [1, 2, 3, 4]. ∎

Since the problem is optimally solved in $\mathcal{P}$ and the algorithm to solve it is well known, we use this result as a baseline for the iterated online exchange. This allows us to compare the matching created by this algorithm along with those created by our online algorithm in the iterated model. In addition, this iteration of the offline optimal is the exact same technique used by `Boardgamegeek`.

4 Iterated Online Exchange

The idea of iterating exchanges over instances where the market evolves allows much richer and dynamic exchanges which certainly influence the mechanism used to run the market. It is also a more realistic setting for a large barter exchange which would likely be repeated and would also likely carries the participants who are not matched to the next exchange. We could explore different kind of iterated exchange, but we focus on *fixed predetermined iterated exchange*, that is the time the exchanges are triggered is known in advanced and are at fixed intervals.

We define the online version of the second-hand matching as having a time interval $[a, b]$ and enforcing to hold trades at least at every interval δ. At any time, participants can get in or opt-out and we are interested in the online optimal exchange possible.

Observation If we restrict the model to only include incoming participant without any participant leaving the market, then there is an upper bound that consist of the optimal matching et the end of the interval $[a, b]$. This bound is trivially proved by the optimality result from last section.

We consider two optimization goals for this problem:

1. Optimizing any weighted sum of the current exchange and the expected next exchange based on a probability model.

2. Optimizing the current exchange and then optimizing the expected next exchange amongst the optimal candidates.

Both goals require a model to estimate the valuation of the next exchange and we define it in the following section, subsection 4.2 defines the expectation-maximization (EM) algorithm we use to solve the first optimization goal. Subsection 4.3 explores the second optimization goal which gives optimality guarantees for the current exchange. We provide experimental results in section 5 based on data from `Boardgamegeek`. Finally, we discuss some of the assumptions and results in section 6.

4.1 Probabilistic Model

We define a probabilistic model of the market in order to be able to compute the expectation step of the EM-algorithm. We constructed the model as an *a priori* evaluation of what the market would probably react; an *a posteriori* analysis based on the setting should be considered for any serious implementation of the techniques we use.

We model the incoming and outgoing participants of each item encountered in the market so far as Poisson distributions, E_i and L_i. For each expected item entering the market we estimate the list of items it can be exchanged for. To do so, we consider two aspects: the general popularity of items wanted in the market, G_i, and the specific popularity of items wanted in the market by those who have traded the exact same item, $S_{i,j}$. The former is used to represent the latent popularity of items in general in the community. We believe this general popularity concept to be present as a whole in the board game community since there is not too many sub-communities which have very orthogonal preferences. The latter, specific popularity, is able to model much finer trends in trades such that a game by a specific designer is often wanted by a player having played previous of his games. We model both of these concepts by a custom discrete probability distribution since we do not known if any *a priori* probability distribution would fit the market we consider. Finally, a simple random variable α associated with a Poisson distribution is used to determine the expected number of game in the wanted list of each item. Table 1 summarize the probabilistic model.

Description	**Var.**	**Distribution**
Number of participants entering the market for item i.	E_i	Poisson
Number of participants leaving the market for item i.	L_i	Poisson
Probability of having item i in W list.	G_i	—
Distribution of G_i on all i.	G	Discrete
Probability of having item i in W_j list.	$S_{i,j}$	—
Distribution of $S_{i,j}$ on all i.	S_j	Discrete
Number of items in the W list.	α	Poisson

Table 1: Summary of the probabilistic model.

We use all the information of the market collected **before** the trade to update the Poisson distributions of the incoming and outgoing items and the expected number of items in the preference list. We use the information collected **after** the trade to update the discrete distributions of the generic and specific popularity based on the actual exchange selected.

4.2 Weighted sum evaluation of the market

In this section, we optimize the weighted sum of the current exchange and the expected next exchange. The algorithm to make the optimization is pretty straightforward based on the probability model and we describe the major steps.

1. Generate the expected items for next round I_{i+1} based on all known E_j and the expected leaving participants $\mathcal{L}$ from all L_j.

2. Pre-generate w different sets of W_k for all $k \in I_{i+1}$ where each W_k is constructed by α draws without replacement from the combined probability distribution $\frac{1}{2}(G + S_k)$. We note $\mathcal{W}_1, \dots, \mathcal{W}_w$, the w different sets.

3. Iterate over all subsets of items in the current market. These subsets $I'_i \subset I_i$ represents the unselected items for the exchange.

    (a) Evaluate the market $I_i \setminus I'_i$ using an offline optimal algorithm.

(b) For each w sets $\mathcal{W}_l$

i. Construct and evaluate the expected market based on the unselected items I'_i, the unmatched items of the evaluation of the current market and the expected market composed of I_{i+1} and $\mathcal{W}_l$.

ii. Compute the running average of the weighted sum of both evaluations of the market for the w sets.

(c) Compare the average with the best encountered so far and store the best one.

4. Execute the current market by excluding the items which produced the best expected outcome found by the algorithm. The current market is executed with the offline optimal algorithm.

Because of step 3, it is clear that our algorithm improves over the iterated version of the offline static optimal as long as our probabilistic model is not horribly wrong. It is also clear that the algorithm runs in exponential time on the number of item in the market because of the same step. We can improve this running time by using a restriction of subsets of size at most k or by using a greedy approach but these technique does not guarantee optimality. Also to really have optimality we would need to search for all optimal solutions at each iteration whereas the above algorithm only finds the first optimal solution it encounters. The next section discusses this variant.

We do not provide additional guarantee because the performance of this algorithm is entirely up to the predictability capacities of the model used since the optimality of the result is at least partly dependent on the expected outcome of the second exchange.

4.3 Evaluation of the market with optimality guarantee

This second optimization goal is intended to settings where it is highly desired to have optimality in the current exchange. Conceptually, the optimization process is executed in two step: first every optimal exchange for the current exchange are found and second, we evaluate each of those in regard to the expected outcome of the next exchange based on the probability model. It is important to note that there are at least some instances where losing an ϵ percentage of optimality in the first exchange could lead to an important improvement in the second exchange. This technique completely overlook this possibility in exchange to a guarantee of optimality.

In order to execute an algorithm capable of implementing this technique, we need to be able to enumerate all optimal matching of the maximum (minimum) matching problem. However, counting the number of optimal matching is $\#\mathcal{P}$-complete, even on a bipartite graph. If we use the variable N_m to represent the unknown number of maximum matching in our graph, then enumerating all maximum matching can be done in $O(|E| \cdot \sqrt{|V|} + |V| \cdot N_m)$ [5].

With some modifications, we can update the previous approach to iterate over these optimal matchings, evaluate the potential futures and select the best average outcome, thereby allowing us to execute this variant. Once again, the predictability of the model is key to selecting the best current strategy but it is certainly better to evaluate those outcomes than to simply pick one optimal matching seemingly at random.

5 Experimental Results

We implemented the first technique, weighted sum evaluation of the market, using the EM-algorithm that we describe earlier. We use two datasets of past exchange from `BoardGameGeek` for evaluation. The exchanges that we use and the number of participants is shown in table 2.

TO10			BENELUX12		
id	**exchange**	**# items**	**id**	**exchange**	**# items**
to_1	Toronto, March 19$^{\text{th}}$ 2009	344	be_1	Benelux, February 2012	446
to_2	Toronto, June 22$^{\text{nd}}$ 2009	298	be_2	Benelux, April 2012	346
to_3	Toronto, September 8$^{\text{th}}$ 2009	312	be_3	Benelux, June 2012	284
to_4	Toronto, November 30$^{\text{th}}$ 2009	634	be_4	Benelux, August 2012	265
to_5	Toronto, March 16$^{\text{th}}$ 2010	618	be_5	Benelux, October 2012	235
to_6	Toronto, June 14$^{\text{th}}$ 2010	287	be_6	Benelux, February 2013	286
to_7	Toronto, September 14$^{\text{th}}$ 2010	541	be_7	Benelux, April 2013	250
to_8	Toronto, November 30$^{\text{th}}$ 2010	1 017	be_8	Benelux, September 2013	411
to_9	Toronto, March 1$^{\text{st}}$ 2011	825	be_9	Benelux, November 2013	297
to_{10}	Toronto, June 7$^{\text{th}}$ 2011	650	be_{10}	Benelux, March 2014	384
			be_{11}	Benelux, August 2014	477
			be_{12}	Benelux, February 2015	667
Total		5 526			4 348

Table 2: Datasets used for the exchanges.

We use equal weights for the current exchange and the expected exchange for the EM-algorithm. We also set the parameter $w = 20$ and try for every subset of unselected participant of size $|I'| \leq 1$. The reason for this low size of subsets is that it greatly increases the computation time. We would certainly achieves better results by increasing this parameter. We run two variants of the EM-algorithm, the first is executed without any prior information about the probabilities of the market and the second is given 3 different trades from 2009 totaling 4795 items to learn the various probability distributions. Table 3 presents the results from our experiment on the TO10 dataset and table 4 presents the results for BENELUX12. We do not include the computation time for both algorithm since the implementation of the EM-algorithm is not optimized whereas the Hungarian method uses a well-known implementation from the optimization module of the external C++ library `dlib`. Nonetheless, the asymptotic running time for the offline optimal is $O(n^3)$ and is $O\left(wn^3 \cdot \sum_{i=0}^{|I'|} \binom{n}{i}\right)$ for the EM-algorithm. With our parameters, the second becomes $O(wn^4)$.

id	**offline algorithm**	**EM-algorithm**		**EM-algorithm (train.)**	
	matched/total	matched/total	% of offline	matched/total	% of offline
to_1	**100**/344	99/344	-1.0%	99/344	-1.0%
to_2	69/542	**109**/543	**57.9%**	77/543	11.5%
to_3	170/785	**204**/746	**20.0%**	175/745	2.9%
to_4	375/1 249	466/1 176	24.2%	**474**/1 212	**26.4%**
to_5	404/1 492	519/1 327	28.4%	**606**/1 375	**50.0%**
to_6	107/1 375	111/1 095	3.7%	**299**/1 072	**179.4%**
to_7	310/1 809	**369**/1 525	**19.0%**	348/1 304	12.2%
to_8	678/2 516	765/2 172	12.8%	**791**/1 915	**16.6%**
to_9	570/2 663	**844**/2 231	**48.0%**	813/1 961	42.6%
to_{10}	359/2 543	387/2 036	7.7%	**676**/1 841	**88.3%**
Total	3 142/15 318	3 873/13 195	23.2%	**4 358**/12 312	**38.7%**

Table 3: Results of the exchanges for the dataset TO10 as described in table 2. Bold values represent the best value per line.

Interpreting the results from both experiment seems to suggest quite heavily that both EM-algorithm performs better than a simple iteration of the offline optimal algorithm. However, the results are not quite as clear whether the training phase is worth doing or not. Both dataset overall were better with the training but not necessarily by a lot. The ambivalence may be because the of the quality of the training data. The variability could also be explained by the fact that the parameter w is too low and a certain luck factor is captured by the random draws. It might also be better to consider the interval of results that those w draws causes to better guide the algorithm toward the "safer" potential future. In any case, it seems that our approach is an improvement over the current implementation of board game exchange even considering the variability concerns.

id	offline algorithm	EM-algorithm		EM-algorithm (train.)	
	matched/total	matched/total	% of offline	matched/total	% of offline
bc_1	**118**/446	**118**/446	**0%**	**118**/446	**0%**
bc_2	93/674	102/673	9.6%	**153**/673	**64.5%**
bc_3	65/865	96/855	47.6%	**151**/803	**132.3%**
bc_4	79/1 065	**152**/1 024	**92.4%**	130/917	64.5%
bc_5	66/1 221	**160**/1 107	**142.4%**	133/1 022	101.5%
bc_6	95/1 441	144/1 233	51.5%	**184**/1 175	**93.6%**
bc_7	97/1 596	**116**/1 338	**19.5%**	92/1 240	-5.1%
bc_8	294/1 910	**332**/1 632	**12.9%**	317/1 558	7.8%
bc_9	67/1 913	55/1 596	-17.9%	**132**/1 537	**97.0%**
bc_{10}	188/2 230	**208**/1 925	**10.6%**	184/1 789	-2.1%
bc_{11}	302/2 519	**397**/2 194	**31.4%**	373/2 081	23.5%
bc_{12}	338/2 884	496/2 464	46.7%	**503**/2 375	**48.8%**
Total	1 802/18 764	2 376/16 487	31.8%	**2 470**/15 616	**37.0%**

Table 4: Results of the exchanges for the dataset BENELUX12 as described in table 2. Bold values represent the best value per line.

6 Discussion

In this section, we discuss some of the assumptions and some results that we obtain. We hope that this discussion provides some insights on the techniques that we use or at least gives some guidance on how to apply similar approaches to different settings.

The probability distribution does not consider for trending changes. In our setting, the release of a very popular game can certainly have an effect on the second-hand market and the discrete probability distribution would have difficulty adapting to those changes. We agree that there is certainly a better way to model this trending effect. On the other hand, we looked at the data in the exchanges and there is a great variety of games, even considering the year of release. In some regards, it also make sense that the second-hand market is not as much influenced by immediate release since it takes time for those new items to reach the market we are concerned about. Nevertheless, we propose two modifications to address this problem:

1. Consider the date that an item entered the market for the first time to properly evaluate the statistics for that particular item.
2. The other option is to only consider the past k exchange for the discrete probability distributions where k is an arbitrary constant. This option

would also solve other problem such as managing the growth of the popularity of the exchange.

The maximal expected outcome takes exponential time to compute. True, and a greedy approach of the problem is not optimal. However, since even the most accurate probabilistic model of the expected next exchange can be wrong, we do not think we should put too much emphasis on optimality in this setting. If some optimality guarantee are to be considered, we strongly recommend using the second technique and only consider the optimal choice for the current exchange. It is up to the model's prediction capabilities and the guarantee requirement to determine whether we should allow more flexibility in the performance of the current exchange compared to the expected next exchange.

Is the data representative of the expected results. We used real data from exchanges made on `BoardGameGeek` which runs the offline optimal algorithm to make the exchange. However, we artificially made one big modification: we carried over the unselected or unmatched participants to the next exchange. In the data from the datasets, some participants who are not matched probably did resubscribed to the next exchange which causes duplicate entries by carrying over the same participants. This effect is certainly a concern for the result that we have, but it is unavoidable from using this particular dataset. On the other hand, we applied the same advantage to both the EM-algorithm and the offline optimal algorithm.

We can also affirm that the good performance of the EM-algorithm compared to the offline optimal is not due to a particularity of the first exchange since we were able to reproduce similar results by starting at the second or third exchange.

The actual exchanges are also quite distanced in time – up to three months. This big interval could make it that the drop out rate would be much higher which we completely neglected in our experiment since we do not have the data to simulate this additional factor.

Finally, since we only considered a single setting, we want to caution against generalizing it to other applications. We expect the result to be much worst in a market with high variability or with low connectivity in the preference graph.

The expectation-maximization technique only try one step into the future. We could not find enough reasons to see if the market could be improved by stepping into multiple steps into the future. We believe the concerns for new trends and optimality of the results would only get worse by foreseeing further ahead. Maybe in some very predictable market this variant could be envisioned but we did not see any signs that this would improve our results in the setting we considered.

7 Conclusion

We have shown that the offline optimal is clearly not always the best technique to use in iterated exchanges for barter trading. It is mostly up to the predictability of the market to determine whether an expectation-maximization approach can provides good results. We showed that the exchange of board games in the second-hand market can be successfully improved by an EM-algorithm. Our algorithm outperforms the offline optimal exchange on almost every exchange except the first one – which is unbeatable by optimality. We experienced an average improvement of 20% to 40% in our two datasets. We also note that it is unclear whether the learning phase is required but it can certainly be useful if the training data is well representative of the expected exchange. Finally, we reaffirm that many of the techniques used in this project can be improved and the discussion section can be a good starting point for some of them.

References

[1] Kuhn, Harold W.(1955): *The Hungarian Method for the assignment problem*83–97.

[2] Munkres, J.(1957): *Algorithms for the Assignment and Transportation Problems*, 1: 32–38.

[3] Tomizawa, N.(1971): *On some techniques useful for solution of transportation network problems*, 2: 173–194.

[4] Edmonds, Jack / Karp, Richard M.(1972): *Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems*, 2: 248–264.

[5] Uno, Takeaki (1997): *Algorithms for enumerating all perfect, maximum and maximal matchings in bipartite graphs.* In: Leong, HonWai / Imai, Hiroshi / Jain, Sanjay (Hg.), Algorithms and Computation.Springer Berlin Heidelberg: 92-101.

7.1 Credit

The title page used in this document is a modification of the front page of the University of Waterloo EThesis template.