Visualizing Match up Matrix

Cong Qiaoben
Stanford University
cqiaoben@stanford.edu

ABSTRACT
In this paper I describe a specific visualization for the match up matrix, which offers insights on the data set that are not available from one dimensional visualization. More specifically, this visualization allows user to perform comparison between each individual by organizing data in meaningful patterns. The whole visualization generation process is automated.

Author Keywords
Human-Centered Computing, Information Visualization

INTRODUCTION
This paper focuses on a specific type of dataset: match up data. The match up data is a 2-dimensional matrix that represents the comparative advantage between players. The comparative advantage is computed based on the win-rate of a certain player when another player is on the other side.

Match up data is common in sports and gaming. However, because the size of the data set, current visualization tends to focus on only one dimension, such as each individual's match up history, top 10 moments, etc.. One significant draw back of an 1-dimensional visualization is the lack of insight on the entire data set. Users could hardly compare between players, not to mention an overview of the data set.

This paper addresses this problem of current visualization methods on a specific data set: Dota2 match up data. Dota2 is a MOBA (multiplayer online battle arena) game. Dota2 is played in matches between two five-player teams, and each player is allowed to pick a hero at the start of the game. This match up dataset consists of the win-rates of each hero against other heroes. Because this data set is quite generic, this visualization technique is not limited to just the Dota2 heroes: it could be applied to the match up data between different teams and players.

To give a concrete example of the problem mentioned above, I will first look at the visualization technique used by dotabuff, a popuplar dota2 analysis website.

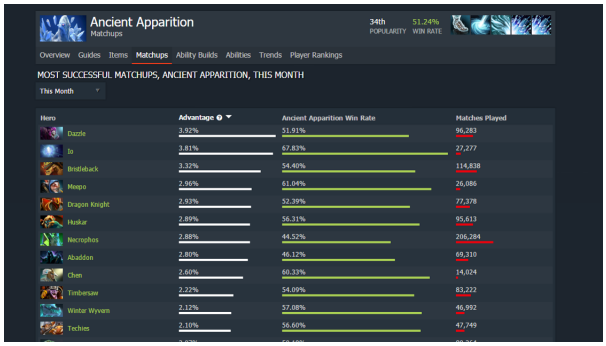

Figure 1. This webpage[1] shows the match up data of a single hero.

There are several problems with this visualization. First of all, as mentioned above, this visual display is quite 1-dimensional. It is rather difficult to compare the match up data between different heroes. Secondly, it is difficult to get a clear overview of how strong this hero is, because this visualization only focuses on using the bars to emphasize each individual data instead of providing a high-level image. Although one could follow the link on the top of this page to see the overall win-rate of each hero, the context switch is not favorable. Lastly, With each comparative advantage displayed in a tabular manner, it is impossible to fit all the data into a single view, and therefore it is hard to compare even between the entries of the same table.

To resolve this problem, this paper will propose a visualization of the entire match up matrix instead of displaying each individual column in a separate view. This matrix will be organized to reveal patterns in the dataset and provide non-trivial insights.

RELATED WORK
Visualizing the entire matrix is a quite established method. The most difficult part of visualizing the entire matrix is to figure out an optimal ordering so that some patterns is emphasized. However, this problem, in general is NP-complete [4]. Therefore, several heuristics were proposed to solve the ordering. One of the earliest approaches use 2D-sort: the idea is to repeatedly sort the matrix based on the weighted sums of columns and rows until the non-zero elements are mostly on the top left corner and lower right corner [4]. Another approach converts the matrix into a bipartite graph,

CHI 2009, April 4 - 9, 2009, Boston, Massachusetts, USA.

with each non-zero entry mapping to an edge in the graph. Then it applies Sugiyamas Algorithm: a heuristic approach to minimize the number of edge crossings of a bipartite graph [4].

Some of the later works focus on the expressing the matrix as clustered heatmap. In fact, using clustered heatmap to visualize the correlation value is quite a common practice among statisticians. *The history of the Clusted Heat Map* offers a clear definition of clustered heatmap:

> The cluster heat map is a rectangular tiling of a data matrix with cluster trees appended to its margins. Within a relatively compact display area, it facilitates inspection of row, column, and joint cluster structure. Moderately large data matrices (several thousand rows / columns) can be displayed effectively on a highresolution color monitor and even larger matrices can be handled in print or in megapixel displays. [7]

The problem is then divided into two sub-problems: the choice of clustering algorithm and the seriation of columns. Several common clustering algorithms include k-means, two-way clustering and hierarchical clustering. [2, 5, 6, 3] The seriation is entirely based on the choice of the seriation loss function, one of the most popular one being the sum of the distances between adjacent rows and columns. [7]

METHODS

The goal of this visualization is to automatically generate a clear visualization that makes the following tasks easy to perform. The first is the comparison between players and the second is player classification. Because those tasks fit quite well in the clustered heatmap, this paper presents an automated way to solve this problem under the idea of clustered heatmap. However, this paper is relatively different from the related work in that the dataset is relatively noisy. Other interesting aspects of this data set will be described later.

Visual Encoding

The visual encoding is red for positive values and green for negative values, with the larger absolute values being less transparent. The first thing I noticed is the use of dots to represent data is better than the use of a standard clustered heatmap when presenting the match up data of 111 heroes. This is because clustered heatmap will have some ”blur” effect when presenting the data, especially when adjacent areas have similar color. This creates confusion when locating a certain entry.

The following image shows an ordered figure of this data set to illustrate this idea.

Although this image (figure 2) does not have a meaningful pattern (other than certain heroes are weak), this image shows the effectiveness of dots in encoding this dataset. The cluster of green dots and red dots could be easily observed. Meanwhile, with high resolution, each individual dot is easily traceable. Therefore, this is a good micro/macro visualization.

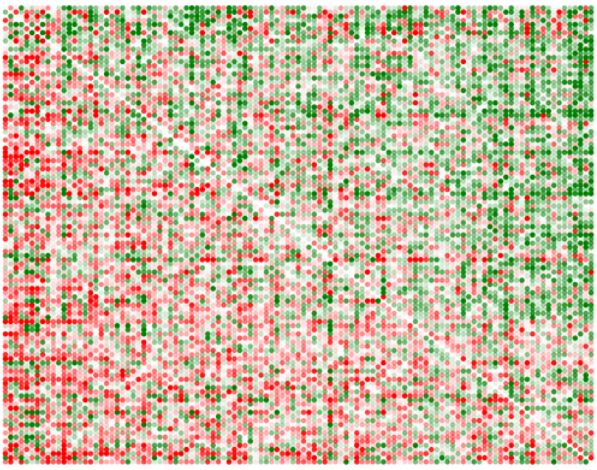

Figure 2. An ordered matrix (dots).

As a comparison heatmap relies heavily on the choice of visual encoding. The following image does not provide an optimal encoding, but that is not the only problem with this image. This image when looked closely creates an uncomfortable low-resolution feeling, even when displayed on a high-resolution screen and it is hard to distinguish the entries.

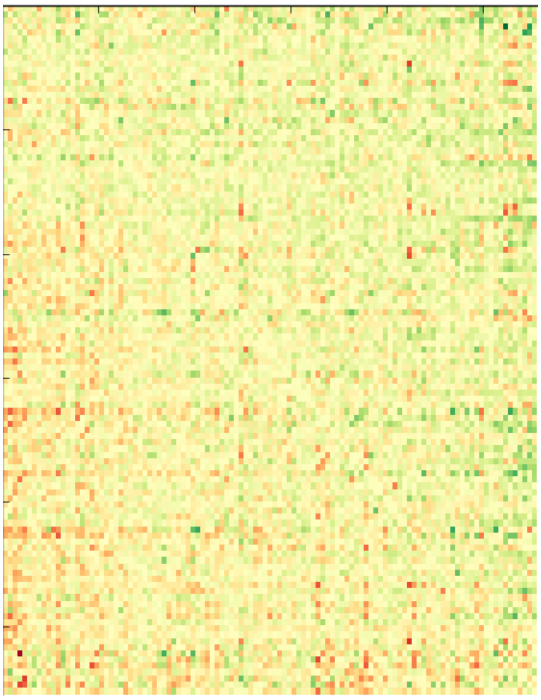

Figure 3. An ordered matrix (standard heatmap).

Therefore, for the following discussion, this paper will use dots instead of color tiles.

Discussion: the match up data set

One interesting aspect of the match up data is it could be clustered. In this specific case, we will expect to see heroes with similar traits to have similar comparative advantage against other heroes. Therefore, if we reorganize the rows and columns of the matrix to have similar heroes close to each other, this will produce a tile pattern.

Therefore, an ideal visualization designed for comparison and classification should be the one that emphasizes the tile pattern. Here the tile pattern shows the visually traceable tiles that consist of dots of similar colors. Thus, a good visualization should emphasize the difference between tiles and the uniformity within each tile. This hints us the seriation loss function should be the sum of the distances within each tile minus the sum of the distances between the tiles.

Another interesting aspect of the match up data is that it is different from correlation values. Therefore, if the heroes are ordered based on the clusters, heroes within the same cluster will be completely noisy. Thus the coloring on the diagonal tiles should look completely random. This contradicts the ideas of the previous work in producing a digonalized pattern.

A side note on this match up data set is its size. The sizes of rows and columns are prohibitively large (111) for any brute force attempt of trying all the permutations. However, with clever reduction, this problem could become solvable by brute force.

The last thing about this data set is it is nearly symmetric. Although the comparative advantages [1] are not symmetric, it is sufficiently close to symmetry. This would become clearer in the later discussion, because it helps emphasizing the tile-pattern.

Framework

This problem as noted could be split in 2 sub-problems, the clustering problem and the seriation problem. The use of the clustering algorithm could be justified by our seriation loss function: we are trying to maximize the distances between adjacent tiles and minimize the distances within a tile. This formulation of the loss function is quite close the one of K-means: we could repeated cluster similar points until all the clusters are sufficiently far away.

Because K-means is a quite popular clustering algorithm, for simplicity we will not discuss it in detail. The potentials for other alternatives will be discussed later.

After using K-means to cluster the data set, the seriation problem could be split into 2 sub-problems as well: the ordering of the clusters and the ordering within a cluster. Notice those two problems are almost independent and are represented by the two aspects of the seriation loss function: the ordering of the clusters influences the distances between adjacent tiles, and the ordering within a cluster influences the distances within a tile. Although the ordering within a cluster, especially the choice of first/last row/column, would influence the distances between tiles, such effect is quite minor considering the distances within cluster is relatively small compared to the distances between clusters. Therefore it is reasonable to ignore such effect in the first step. Later this paper will show that the solution to the second problem also provides an optimal solution to emphasize the distances between the tiles by ordering heroes within a cluster.

Results: K-means approach

The following results will illustrate that this data set fits well with K-means. The number of clusters used here is 10. This is picked based on an estimation of clusters within this data set, as well as the computation limit of the following steps, which would be described later.

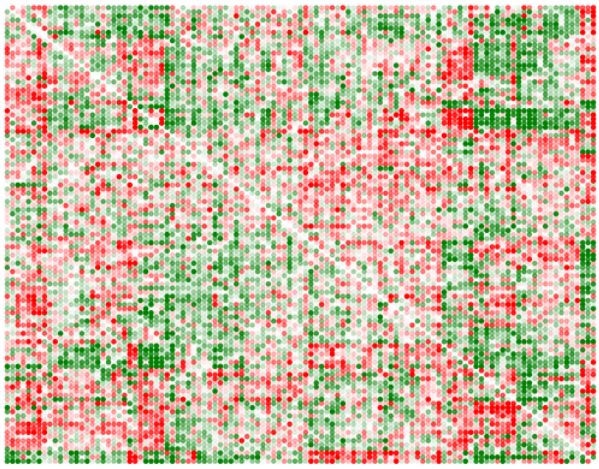

Figure 4. K-means result.

The figure above does not use any of the seriation heuristics. It just chooses a random ordering as long as heroes in the similar clusters are placed in the same range. The result happens to be an effective visualization because of a clear tile pattern.

Notice there is a line of blank dots on diagonal. This is because the ordering of rows and columns are the same, and the comparative advantage of an hero against itself is undefined. If the users look closely, they will observe that this matrix is nearly symmetric. Again, this is because the ordering of rows and columns are designed to be the same to emphasize the symmetry.

In fact, the near symmetric pattern in the data set plays a huge role in emphasizing the tile pattern. This design intentionally violates the effective data ink theory, which emphasizes the importance of effectively display the data with as little ”ink” as possible. Clearly the near symmetry pattern here indicates some kind of redundancy. However, one reason I choose this approach is that the data set is not entirely symmetric, so it does not make sense to remove half of the data. Another aspect, which is more important, is the illustration of the tile pattern.

This following image shows the result of removing such redundancy. The tile pattern is not as obvious as in the figure above.

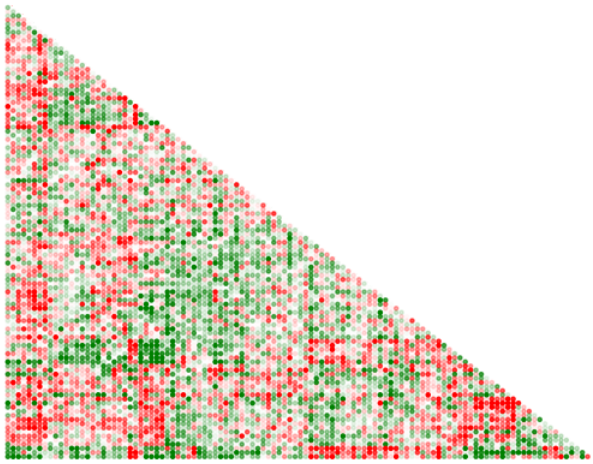

Figure 5. K-means result with highest data/ink ratio

The following image shows a reasonable interpretation of figure 4.

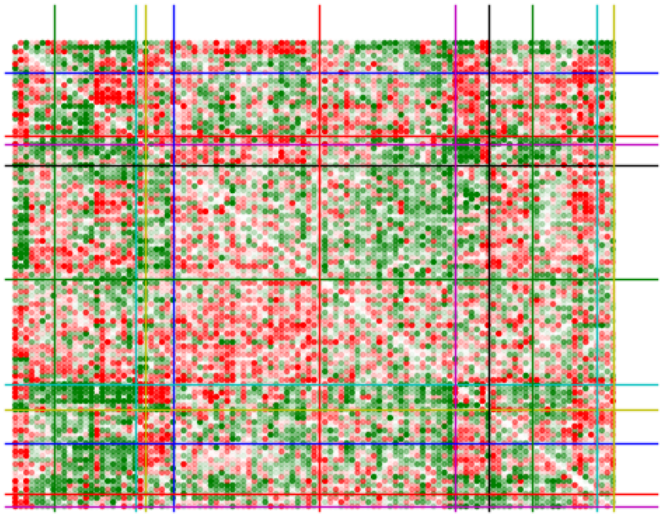

Figure 6. K-means result (each line separate clusters).

This, compared with figure 2 which does not show any tile patterns, tells us that K-means should be relatively good at emphasizing the difference between tiles. However, we could not rely completely on the randomness in producing a nearly perfect tile pattern.

Seriation problem: step 1

The results above show that K-means algorithm is applicable in this case. However, because we could not rely entirely on the randomness, the introduction of an effective seriation loss function for the ordering of clusters is necessary.

Because the objective is to maximize the differences between adjacent tiles, it makes sense to use the sum of the distances between adjacent tiles as the loss function. Because each tile tends to give us a feeling of green/red, it makes sense to use the average of each tile to encode this perception.

Another possible heuristic is based on the users' tendency of focusing on the diagonal. Therefore the loss function is this case will be the sum of absolute values of the average of the tiles, with each value weighted the inverse of the distance to diagonal.

Results: step 1

The clustering algorithm reduces the dimension from 111 to 10. This allows a brute force approach to iterate through all possible permutations to find the optimal solution (the run time is not too computationally heavy, roughly 30 secs) The following images shows the optimal results.

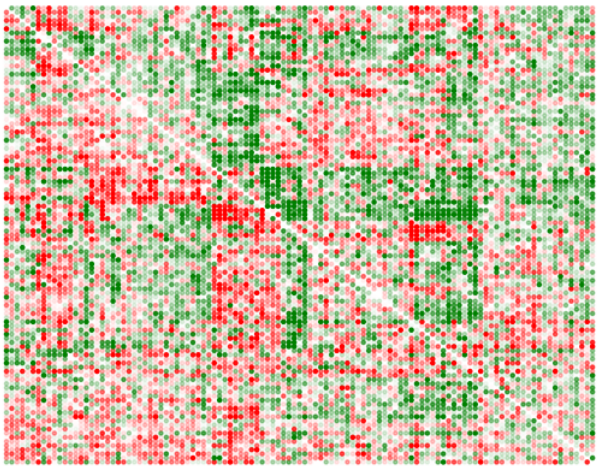

Figure 7. Emphasizing tile pattern as well as diagonal. Final result

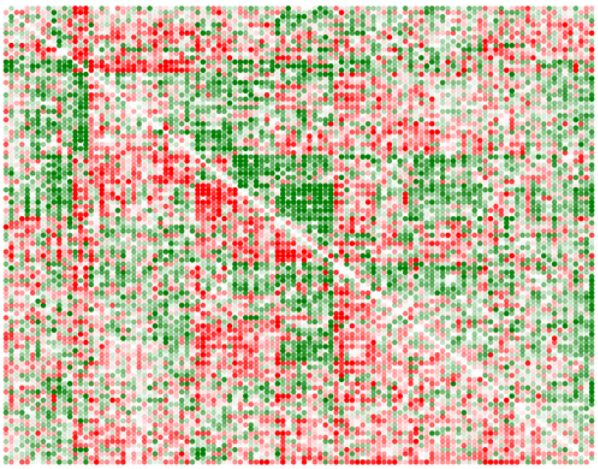

Figure 8. Emphasizing diagonal

Most users tested on this pair of images reported that the tile pattern is clearer in the first image, although the second image looks "favorable". However, all the users say figure

7 is better than figure 4. The user feedback is reasonable because people tend to favor images with more weights on diagonal.

The reason figure 7 is better than figure 8 is that while the width of the stripes are smaller compared to that of figure 8, it is more accurate. Figure 8 sometimes put similar tiles in adjacent positions, and thereby produces an illusion of large tiles that actually consists of several smaller tiles.

Seriation problem: step 2

The step 2 uses the idea of sorting. Using the algorithm mentioned in the related work [4], the result reduces the distances within clusters. Moreover, one side effect is the visual enhancement of the edges between the tiles. Because red dots tend to move towards upper left and green dots tend to move towards lower right, we will see more contrast on the "green-red" edges (green tile on the left and red tile on the red).

Results: step 2

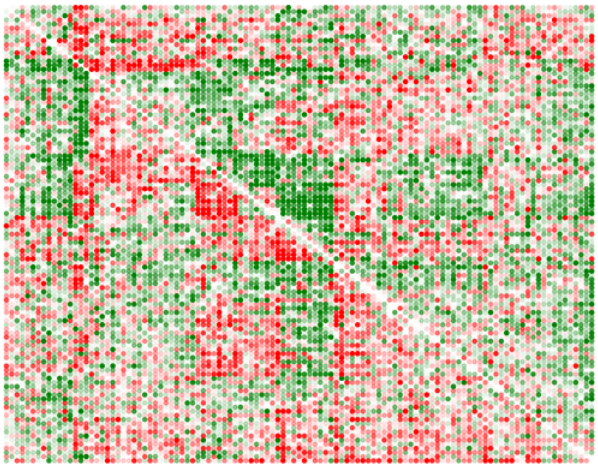

Figure 9. Emphasizing diagonal, each tile sorted

The result is not so obvious in this case when comparing figure 9 to figure 8. This is because although figure 8 does not arrange the dots within each cluster in an optimal fashion, it does visually produce edges. The problem is the edges are sometimes at the wrong places and it is hard to detect. Therefore, those two matrices produces the same "shape", but the edges of the first one is off by a few pixels. This effect could be observed only when those two figures are examined carefully.

Discussion

This visualization proposes an easy way of classifying and comparing players/heroes/teams by revealing a clear tile pattern. The final (optimal) result is figure 7. This paper shows that it is possible to fulfill those tasks, with good clustering algorithm and seriation loss function. As noted, the choice of the clustering algorithm is completely up to the user, as long as the cluster algorithm produces a result that maximizes the distances between clusters and minimizes the distances within clusters.

Future work

One significant drawback of this visualization is it is hard to label the data. However, allowing interaction could mitigate this problem. If the data set does not (significantly) update very often, this visualization could be easily extended into a query system, with users typing in keywords and corresponding dots got highlighted on the matrix.

Another problem with the current algorithm is that it uses brute force to solve the optimal solution. Although the brute force attempt shows that such algorithm is applicable to the match up data set, in practice 30 secs would be quite unreasonable for a good interactive system. Thus another future direction would be trying out different heuristics that could be solved in polynomial time or use some randomized approach.

Another possible extension is to have a different loss function for distances between a tile that looks completely random and a tile that looks green/red. Because on a higher level a random tile should not be treated the same as a green/red tile.

REFERENCES

1. Ancient Apparition. `http://www.dotabuff.com/heroes/ancient-apparition/matchups`.
2. T. City. Visualizing Team Units Using Hierarchical Clustering. `https://thecity2.wordpress.com/2012/03/02/visualizing-team-units-using-hierarchical-clustering/`.
3. J. Kettenring. The Practice of Cluster Analysis.
4. E. Makinen and H. Siirtola. Reordering the Reorderable Matrix as an Algorithmic Problem. `http://www.mat.ucsb.edu/~g.legrady/academic/courses/15w259/d/re_orderableMatrix.pdf`.
5. A. B. Morris, S. A. and G. G. Yen. Dendrogram seriation using simulated annealing. *Information Visualization*, (2), 2003.
6. J. Weinstein. A Postgenomic Visual Icon. *Science*, (319), 2008.
7. L. Wilkinson and M. Friendly. The History of the Cluster Heat Map. 2008. `https://www.cs.uic.edu/~wilkinson/Publications/heatmap.pdf`.