RELAXED GRAPH MATCHING FOR ANALOGICAL REASONING

Simon Segert
Princeton Neuroscience Institute
Princeton University
Princeton, NJ
ssegert@princeton.edu

Mariano Tepper
Intel Labs
Intel Corporation
Hillsboro, OR
mariano.tepper@intel.com

Javier Turek
Intel Labs
Intel Corporation
Hillsboro, OR
javier.turek@intel.com

Jonathan Cohen
Princeton Neuroscience Institute
Princeton University
Princeton, NJ
jdc@princeton.edu

ABSTRACT

Analogical reasoning has classically been modeled using a graph matching framework. However, this problem is known to be computationally intractable, raising the question of finding approximate solution methods that perform well on the sorts of varied domains in which humans excel. In this work, we propose such an algorithm that is based on a continuous relaxation of the graph matching objective. We show the algorithm is capable of knowledge transfer in the domain of abstract multi-step reasoning tasks.

1 INTRODUCTION

Analogical reasoning, defined as the ability to transfer knowledge from one domain to another, is a crucial component of the ability to generalize out-of-domain (Gentner, 2003), an ability in which machines still lag behind humans. Modern approaches to analogy in artificial systems, such as variational autoencoders (VAEs), (Chen et al., 2018; Higgins et al., 2017) or word embeddings (Mikolov et al., 2013) often rely on strong assumptions about the structure of the underlying domain, such as the existence of linearly disentanglable representations in the former, or of parallelogram structure in the latter. By contrast, classical models of human analogy, such as Gentner's Structure Mapping Theory (SMT) (Gentner, 1983), allow for domains of essentially arbitrary complexity, albeit the basic objective becomes computationally intractable in general. In this work, we propose an approach intermediate between these two extremes: capable of handling non linearly-separable data such as the SMT approach, but also differentiable and implementable by a neural network architecture such as those used in modern approaches to machine learning problems.

To do this, we assume that we are operating on a graph-structured domain. Many naturally occurring domains can be cast in this way. For example, navigation and learning problems can be described in terms of Markov Decision Processes (Sutton & Barto, 2018) in which the graph is composed of possible states connected by the allowed actions. Similarly, conceptual knowledge can be described in terms of knowledge graphs (Gentner, 1983), in which the edges encode various kinds of relations between concepts. Complementing these approaches, recent work in neuroscience has suggested that certain graph-structured domains are represented in the brain via a successor representation (Dayan, 1993), which encodes the graph in an approximate vectorized form (Stachenfeld et al., 2017). Crucially, our approach allows for such a representation and does not require exact specification of the underlying graph structure.

Following Gentner (1983), we formulate analogical inference as a graph matching problem. In this classical computer science problem, we seek to establish a map between vertices in a source graph to vertices in a target graph. Our key contribution lies in moving down the Marrian hierarchy (Marr & Poggio, 1976) from the Computational level to the Algorithmic level, by proposing a computationally tractable neural network architecture. Our approach yields approximate solutions using an appropriate continuous relaxation (Fiori et al., 2013; Vogelstein et al., 2014), allowing it to gracefully handle the kinds of inherently-nonlinear structural forms that commonly occur in the environment (circles, trees, etc.).

Once we have matched the graphs, we gain the ability to solve simple 4-term analogies "for free": to complete an analogy of the form "A is to B as C is to what?", we simply have to see where B gets sent in the obtained matching. However, learning a matching also allows us to transfer sequences of points from one domain to another, rather than just a single point as in a classical analogy. We show how this can be leveraged to perform multi-step "planning by analogy" in the domain of abstract problem solving.

2 A NEURAL NETWORK FOR GRAPH MATCHING

Assume we are given graphs G_1 and G_2, with vertex sets denoted $V_1 = \{v_1^1, \ldots, v_{|V_1|}^1\}$, $V_2 = \{v_1^2, \ldots, v_{|V_2|}^2\}$ and edge sets denoted E_1, E_2. The respective adjacency matrices are denoted $\mathbf{A}_1$ and $\mathbf{A}_2$. Note that in general $|V_1| \neq |V_2|$. Furthermore, we suppose that G_1 and G_2 contain certain distinguished points, the images of which are constrained a priori under any mapping. Formally, let $X = \{(v_{i_k}^1, v_{j_k}^2)\}_{k=1}^m$ be a collection of m ordered pairs of vertices in G_1 and G_2. We refer to these as Marked Points. Here $1 \leq i_k \leq |V_1|$ and $1 \leq j_k \leq |V_2|$ for all k. [1] As above, the marked points could correspond to the "A" and "C" terms of an analogy; we will also see examples in Sec. 3 that naturally involve multiple pairs of marked points.

Following the basic Structure-Mapping theory, we pose the problem of finding a function $F : V_1 \to V_2$ that respects the connectivities of the two graphs as nearly as possible. That is, we want to minimize the number of pairs $(v, v') \in V_1 \times V_1$ for which the proposition $(v, v') \in E_1 \Leftrightarrow (F(v), F(v')) \in E_2$ is violated. We also require that $F(v_{i_k}^1) = v_{j_k}^2$ for every k. A straightforward argument show that for any such function F, the number $N(F)$ of such violating pairs is given by

$$N(F) = \|\mathbf{Q}_F \mathbf{A_2} \mathbf{Q}_F{}^\top - \mathbf{A_1}\|_1. \tag{1}$$

The matrix is defined as $\mathbf{Q}_F[i,j] = 1$ if $F(v_i^1) = v_i^2$ and 0 otherwise, and the $L1$ norm of a matrix is defined as the sum of absolute values of its entries. A detailed derivation is given in Supplement D. Minimizing N is known to be NP-hard (Loiola et al., 2007).

Note that $\mathbf{Q}_F$ is a quasi-permutation matrix, meaning that every row has exactly one entry equal to 1, with the rest being 0. In particular, all entries are non-negative and each row sums to 1. A matrix with these properties is called Stochastic. This suggests that the problem can be "relaxed" by dropping the constraint that each entry of $\mathbf{Q}_F$ be either 0 or 1, and requiring only that the matrix be stochastic (Lyzinski et al., 2015; Vogelstein et al., 2014). We denote the set of stochastic matrices $\mathbf{P}$ of shape $|V_1|$ rows by $|V_2|$ columns such that $\mathbf{P}[i_k, j_k] = 1$ for all k by SM_X.[2] This leads us to consider the loss function:

$$L_X(\mathbf{P} \mid \mathbf{A}_1, \mathbf{A}_2) = \|\mathbf{P}\mathbf{A}_2\mathbf{P}^\top - \mathbf{A}_1\|_1, \mathbf{P} \in SM_X. \tag{2}$$

The vertical bar notation is meant to indicate that $\mathbf{A}_1$ and $\mathbf{A}_2$ are regarded as inputs and can be replaced with other matrices of the appropriate sizes. We will discuss this further in the next section.

In contrast to the set of mappings $V_1 \to V_2$, SM_X is a continuous space, so we can use gradient-based techniques to perform the optimization. This is done by parametrizing the space of stochastic matrices with a simple neural network, with details given in Supplement E [3].

Next, there is the matter of obtaining a valid mapping once we have solved the relaxed version . This can be done in a variety of ways; here we will do so by simply taking a row-wise argmax of $\mathbf{P}$. More explicitly, given a solution $\mathbf{P}$ to the relaxed problem, define the associated mapping $F_\mathbf{P} : V_1 \to V_2$ by:

$$F_\mathbf{P}(v_i^1) = v_{\arg\max(\mathbf{P}[i])}^2, \tag{3}$$

where $\mathbf{P}[i]$ denotes the ith row of the matrix $\mathbf{P}$.

[1] In order to rule out inconsistent constraints, we require that $i_k \neq i_l$ for $k \neq l$.

[2] In general, we use the letter $\mathbf{Q}$ for quasi-permutation matrices, and letter $\mathbf{P}$ for stochastic matrices.

[3] Note that we solve each matching problem online, rather than employing a train-test paradigm.

Variants. While the above section dealt with the adjacency matrices of the graphs, the interpretation is not fundamentally changed if these are replaced with other types of affinity matrices on the vertices. Of particular interest to us is the successor representation $\mathbf{SR}_G = (\mathbf{1} - \gamma\overline{\mathbf{A}_G})^{-1}$, where $0 < \gamma < 1$ is a parameter and $\overline{\mathbf{M}}$ denotes the operation of normalizing each row of matrix $\mathbf{M}$ to sum to 1. As alluded to in Sec. 1, this matrix has a well-known interpretation in the context of reinforcement learning (Dayan, 1993). This leads us to consider the modified objective function $L_X(\mathbf{P}\,|\,\mathbf{S}_1, \mathbf{S}_2)$, where $\mathbf{S}_i = \mathbf{SR}_{G_i}$ and $\mathbf{P} \in SM_X$ as before.

Secondly, we consider a variant that often performs better in practice. Taking inspiration from the definition of a normalized graph Laplacian,we consider the modified objective:

$$\overline{L}_X(\mathbf{P}\,|\,\mathbf{A_1}, \mathbf{A_2}) := \|\overline{\mathbf{P}\mathbf{A}_2\mathbf{P}^\top} - \overline{\mathbf{A}_1}\|_1, \mathbf{P} \in SM_X, \tag{4}$$

where, as above, $\overline{\mathbf{M}}$ denotes the matrix obtained by normalizing each row of the matrix $\mathbf{M}$ to sum to 1. These two modifications can be applied in tandem (i.e., replace the adjacency matrix with the SR and normalize both terms), with the corresponding loss denoted $\overline{L}_X(\mathbf{P}\,|\,\mathbf{S}_1, \mathbf{S}_2)$. In what follows, we will always assume the loss function to be of this form and, for the successor representation, $\gamma = .8$ unless otherwise specified.

Finally, the effect of different relaxations than in Eq. (2) is discussed in Supplements B and C.

3 Applications to knowledge transfer

We now formalize how the model may be used for knowledge transfer of multistage planning. Suppose we have two state graphs G_1 and G_2 corresponding to two different planning or reasoning tasks. These tasks come with distinguished nodes $s_1, t_1 \in V_1$ and $s_2, t_2 \in V_2$ corresponding respectively to the starting and target states of each task. Finally, we suppose that we have a solution to the first task, namely a path of vertices $s_1, v_1, \ldots, v_k, t_1$ of vertices on G_1 from the initial state to the target state. To transfer our knowledge, we then solve the matching problem from $G_1 \to G_2$, subject to the constraints that t_1 maps to t_2 and s_1 maps to s_2, in the way described in Sec. 2. We then obtain a candidate solution to G_2 by simply mapping each vertex v_i on the path according to the solution matrix $\mathbf{P}$. That is, we obtain a candidate path $\tilde{v_1}, \ldots, \tilde{v_k}$ from s_2 to t_2 given by $\tilde{v_i} = F_{\mathbf{P}}(v_i)$, with $F_{\mathbf{P}}$ as in section 2. Note that, a priori, this need not be a valid path on G_2 (i.e. successive vertices might not be connected); however we will see that when the graphs have common structure the paths obtained in this way do indeed tend to be valid.

As a first illustration, we consider the well-known Tower of Hanoi task, in which the player is presented with a set of disks, each of a different size and distributed over some number of pegs, and given the goal of moving all of the pegs to a target disk (the target state) under the constraint that only one disk can be moved at a time (to any peg) and a larger disk can never be placed over a smaller one on a given peg. For simplicity, we consider two cases, both of which have three disks and three pegs. In the first case, all of the disks start on the leftmost peg, and the goal is to move them all to the rightmost peg. Second, we consider a simple variant in which the objective is instead to move all disks from the first peg to the middle one. In both cases, a graph can be described with a node for each valid configuration of the disks, and edges corresponding to valid moves. In the notation used above, the two graphs G_1 and G_2 are identical, and the initial states s_1 and s_2 coincide, but the target states t_1 and t_2 are different. As shown in Fig. 1, the algorithm discovers a mapping that carries a solution to the first task into a valid solution of the second.

As a slightly more complex example, we let G_1 be the state graph corresponding to the "Farmer/Goat" problem and G_2 that of the "Missionaries and Cannibals" problem. We give precise descriptions of these puzzles in Supplement F. These are both "river crossing" puzzles in which the objective is to transport various parties across the river using a small boat, subject to constraints about which parties can be left alone with each other. Similar to the previous example, the nodes of the graphs correspond to valid configurations of the various parties on the two shores. It was shown in (Holyoak & Thagard, 1989) that children who had first learned how to solve the simpler Farmer/Goat problem were were more successful in finding a solution to the more complex Missionaries/Cannibals problem. However, no such transfer was obtained in the opposite direction.

In Fig. 2 (shown in Supplement A) we show how a matching from the Farmer/Goat graph to the Missionaries/Cannibals graph indeed carries a solution of the former to a solution of the latter (in

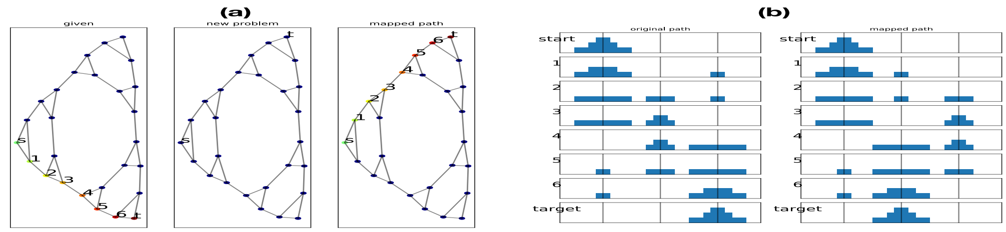

Figure 1: Mapping a solution to the original Tower of Hanoi task to a solution to the modified task. (a) Left: Original task with solution path indicated. Note “s” and “t” stand for “start” and “target” respectively. Middle: Modified task. Right: The image of each point on the path in the left image under the obtained mapping. (b) Rendering of the states in the original and mapped path from part (a). (Figure best viewed in zoom).

fact there are two solutions to the Farmer/Goat problem, both of which have this property). However, as with people, the matching obtained in the opposite direction is of low quality. Indeed, of the 768 possible shortest solution paths to the Missionaries/Cannibals problem, only 4.7% are mapped to valid solutions of the Farmer/Goat problem. An example solution path obtained by this mapping is shown in Fig. 3 (in Supplement A).

Finally, we provide a different type of example that shows how our algorithm can exploit state space abstraction for efficient problem solving, by relating the simpler, abstract representation of a state space to a more complex but veridical representation of that space. Often problems are solved hierarchically: one first solves a suitably abstracted version of the problem and then “fills in the details” when solving the actual problem (this can be repeated at several levels of abstraction) (Botvinick, 2012). Spatial navigation tasks provide an example: if one’s goal is to travel from New York to Los Angeles, then the initial planning stage would probably consist of steps like “Drive to JFK, board plane, ...” etc. rather than “Move car into left lane on 2nd Avenue to merge onto I-495 East, ...” etc. In order to implement such a strategy, it is necessary to have a correspondence between the abstracted state space and the original.

As an example of finding such a correspondence, we return to the Tower of Hanoi. The state graph for this problem has a recursive structure: the graph for n disks can be obtained from the graph for $n-1$ disks by replacing each node in the smaller graph with a clique of size 3. We see in Fig. 4 (shown in Supplement A) that the algorithm is able to reveal this structure in the course of mapping the larger graph onto the smaller one. We can interpret this mapping as a coarsening of the larger graph such that the groups correspond exactly to these cliques.

4 Discussion

We have described a differentiable algorithm that is capable of structure mapping in wide range of problems, and illustrated its use in multi-step analogical reasoning and problem solving. The algorithm draws inspiration both from cognitive psychology (representing problem spaces) and mathematical optimization (approximating solutions to combinatorial problems). Many problems that people must solve, such as language processing (Barton et al., 1987) and spatial navigation (Macgregor, 2011), are intractably combinatorial; our approach gives a guide for how approximate solutions to them may be found by mapping known structure to novel situations. This exploits the fact that, although graph matching is NP hard in the worst cases, not all *cases* are necessarily so. We argue that the kinds of problems that people encounter and are able to solve are not the “worst case” ones, but rather closer to the structured examples considered in this paper. Our method is thus parsimonious in that it (1) works well in these structured cases and (2) these structures are not explicitly “built in” to the algorithm itself. Furthermore, our approach complements recent work on graph similarity learning,in particular Li et al. (2019), in which the authors use a graph network to learn a scalar matching-based similarity measure, rather than an explicit node-wise correspondence as we do.

Finally, there are several possible extensions that we plan to address in future work, for example extension to the case in which the solution paths have different lengths, and a more systematic treatment of graphs with recursive or self-similar structure.

REFERENCES

G Edward Barton, Robert C. Berwick, and Eric Sven Ristad. *Computational Complexity and Natural Language*. MIT Press, 1987.

Matthew Michael Botvinick. Hierarchical reinforcement learning and decision making. *Current opinion in neurobiology*, pp. 956–962, 2012.

Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems*, 2018.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 1993.

Marcelo Fiori, Pablo Sprechmann, Joshua Vogelstein, Pablo Muse, and Guillermo Sapiro. Robust multimodal graph matching:sparse coding meets graph matching. *Advances in Neural Information Processing Systems*, 2013.

Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7: 155–170, 1983.

Dedre Gentner. Why we're so smart. *Language in mind: Advances in the study of language and thought*, 2003.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. *International Conference on learning Representations*, 2: 891–921, 2017.

Keith Holyoak and Paul Thagard. Analogical mapping by constraint satisfaction. *Cognitive Science*, 1989.

Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. *arXiv:1904.12787.*, 2019.

Eliane Maria Loiola, Nair Maria Maiade de Abreua, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176, 2007.

Vince Lyzinski, Donniell E. Fishkind, Marcelo Fiori, Joshua T. Vogelstein, Carey E. Priebe, and Guillermo Sapiro. Graph matching: Relax at your own risk. *IEEE transactions on pattern analysis and machine intelligence*, 38, 2015.

James N. Macgregor. Human performance on the traveling salesman and related problems: A review. *Journal of Problem Solving*, 2011.

David Marr and Tomaso Poggio. From understanding computation to understanding neural circuitry. *Artificial Intelligence Laboratory. A.I. Memo. Massachusetts Institute of Technology*, 1976.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013.

Kimberly L Stachenfeld, Matthew M Botvinick, and Samuel J Gershman. The hippocampus as a predictive map. *Nature neuroscience*, 20, 2017.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Joshua T. Vogelstein, John M. Conroy, Vince Lyzinski, Louis J. Podrazik, Steven G. Kratzer, Eric T. Harley, Donniell E. Fishkind, R. Jacob Vogelstein, and Carey E. Priebe. Fast approximate quadratic programming for large (brain) graph matching. *arxiv:1112.5507v5*, 2014.

Supplementary Material

A Additional figures

Fig. 2 shows the mapping of the Farmer/Goat problem onto the missionaries and cannibals problem. Here each solution of the simper problem is mapped to a valid solution of the more complex problem. Fig. 3 shows the image of an example solution path under the matching in the reversed direction. Unlike in the previous case, the image is not a valid solution path.

Fig. 4 shows the mapping of the Hanoi graph onto the graph of the 2-disk variant. We see that the mapping reveals the recursive structure by mapping all points in the indicate cliques to the corresponding nodes on the smaller graph.

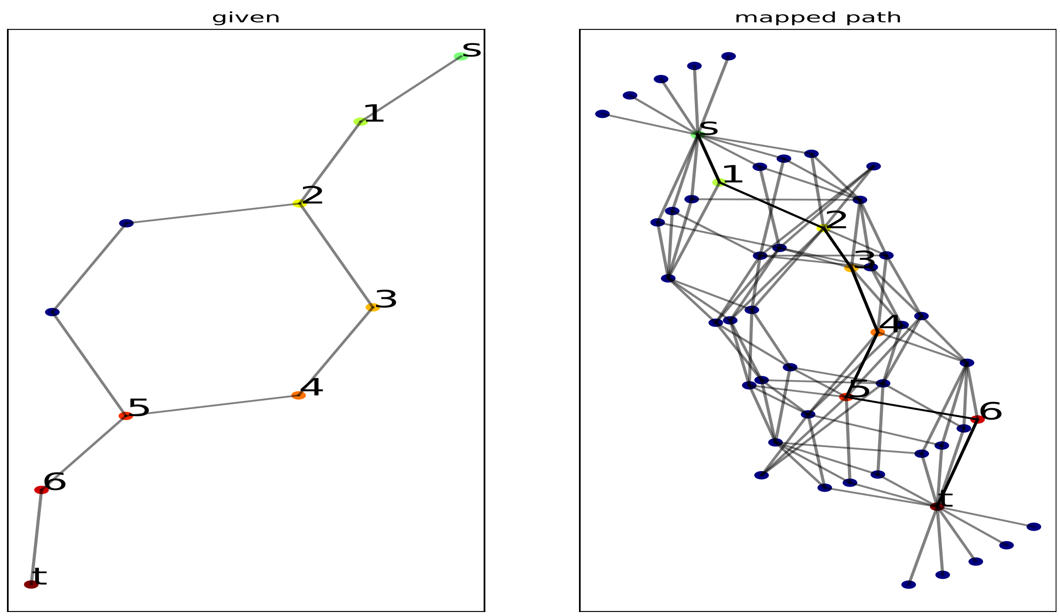

Figure 2: Left: The state graph for the Farmer/Goat problem with a solution path indicated. Right: The state graph for the Missionaries/Cannibals problem, with the images of the solution path indicated. Note that the mapped path is a valid path on the target graph (and the same is true of the alternative solution path). Note that the mapping here uses the adjacency matrices rather than the Successor Representations. As in Figure Fig. 1, "s" and "t" denote the starting and target states in each problem.

B Comparison with indefinite Frobenius relaxation

In (Lyzinski et al., 2015) a similar matching problem is considered. The authors prove optimiality results for the objective function $||\mathbf{P}\mathbf{A}_2\mathbf{P}^T - \mathbf{A}_1||_F^2$, $P \in DSM$, with DSM denoting the space of doubly-stochastic matrices and the subscript F denoting the Frobenius norm of a matrix. They term this the "Indefinite loss function". In view of their theoretical results, it is interesting to compare our L1 loss with theirs.

Note that their assumptions differ from ours in the following ways: (1) they assume a particular random graph model for G_1 and G_2, while we consider highly structured graphs and (2) they assume G_1 and G_2 to have the same size.

In particular, in our setup, it is impossible to use the space of doubly stochastic matrices. This is because we do not assume the graphs to have the same size, and a doubly stochastic matrix is necessarily square. However, if we drop this assumption and assume P to be only row-stochastic, then we can use their loss function as a drop-in replacement for ours.

In the next sections, we directly compare the obtained results with our own loss function.

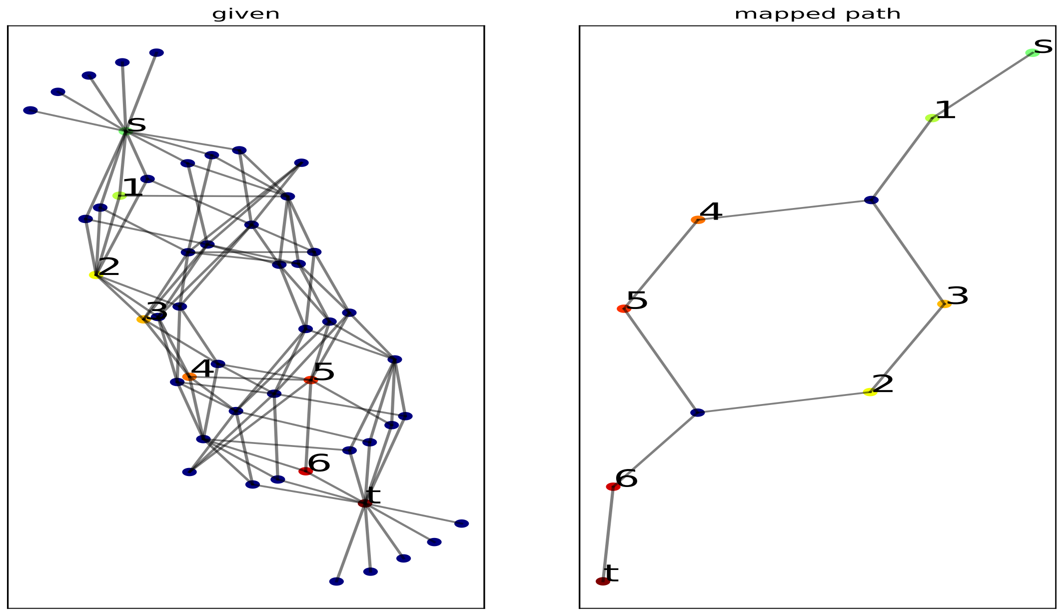

Figure 3: Similar to Fig. 2, but for the mapping in the reverse order. An example solution path is shown for the Missionaries/Cannibals problem on the left, and the corresponding mapped path to the Farmer/Goat problem on the right. Note that the mapped path is not a valid solution to the Farmer/Goat problem.

B.1 Figures with Frobenius norm

Here, we use the same figures as in the main text and Supplement A , but using the "indefinite loss" $\|\mathbf{P}\mathbf{A}_2\mathbf{P}^T - \mathbf{A}_1\|_F^2$ as in (Lyzinski et al., 2015). In Fig. 5, Fig. 6 and Fig. 7, we reproduce the settings of Fig. 1, Fig. 2 and Fig. 4, except using the indefinite loss in place of our L_1 loss. We see that the matchings obtained in Fig. 5 and Fig. 7 of worse quality compared to their L1 counterparts, with the results for the Farmer/Goat in Fig. 6 being similar to the L1 case.

C Results on Lines, Circles, Trees

We consider the problem of solving simple 4-term analogies on model linear and non-linear spaces. In what follows, let G be either a directed linear graph, a directed circle graph, or a directed binary tree. Denote by d_G the undirected geodesic distance function on G (i.e., the length of the shortest undirected path between two points). Denote by S_γ the successor representation of G, defined using the directed adjacency matrix of G. As in the main text, we set $\gamma = .8$ in what follows.

On these experiments, we used a binary tree with 63 vertices, a linear graph with 31 vertices, and a circle with 31 vertices and 100 analogies for each graph. Each matching problem is solved using 10 random intializations.

C.1 Generating analogies

To generate analogies, we choose A, B, C randomly from the vertices of G subject to the constraints that $A \neq B$ and $A \neq C$, $d_G(A, B) \leq M$ and the analogy has at least one valid answer. Here M is set to 4 for the tree and 8 for the other two graphs.

In the case of lines and circles, each analogy has an unambiguous correct answer, denoted by D_{true}. In the case of a tree, an analogy may have several acceptable answers. For example, if A is a parent of B, then either child of C would be an acceptable answer to the analogy. In general, we consider D to be correct if (1) $d_G(A, B) = d_G(C, D)$ and (2) the shortest path from C to D has the same pattern of forward/backtracking with respect to the edge directions on G as the shortest path from A to B. We thus propose the following unified measure of analogy quality on the three graphs: Consider the shortest undirected path from A to B. Associated to this path is a binary sequence

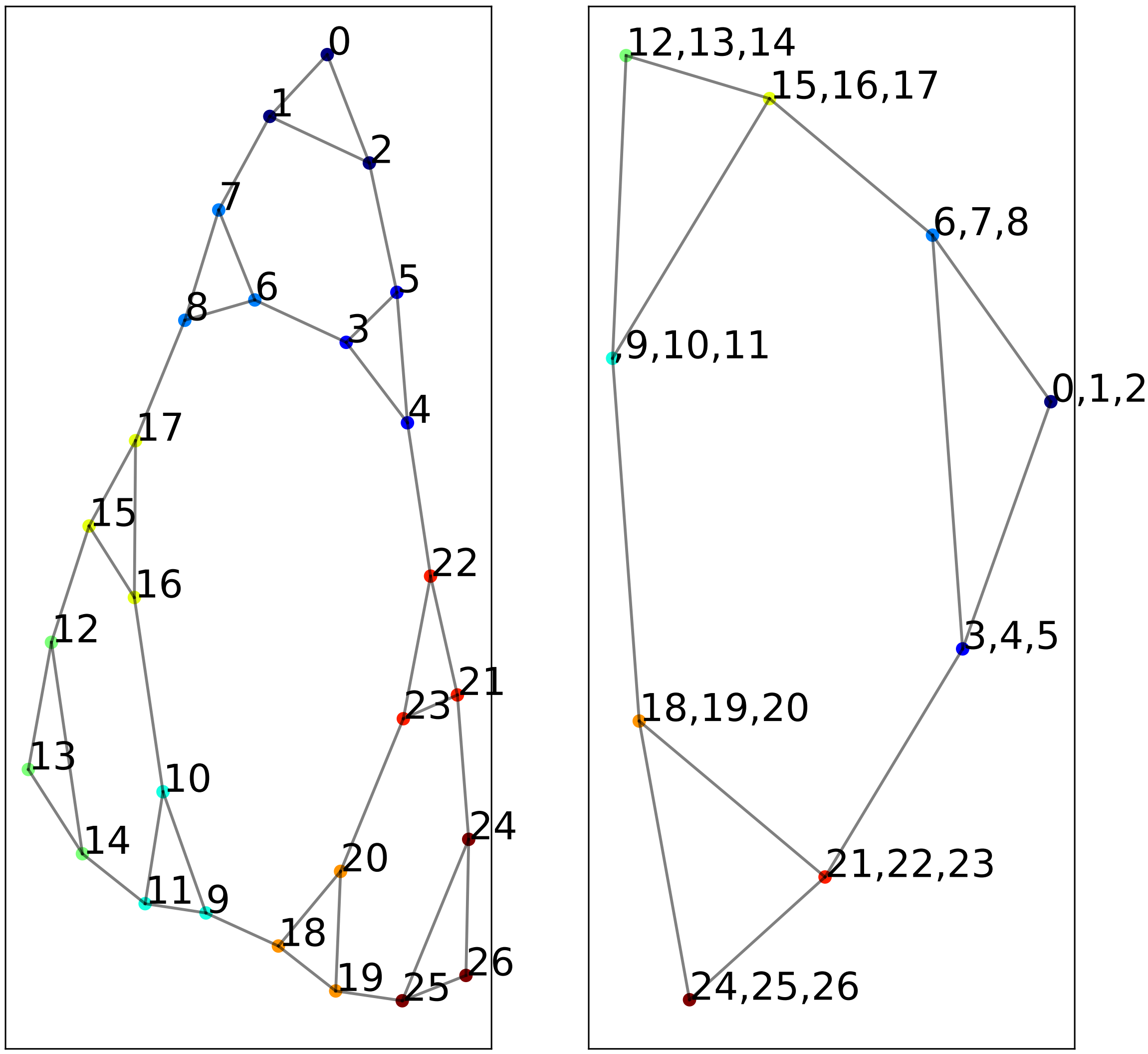

Figure 4: Mapping the state graph for Towers of Hanoi (left) to that of two-disk variant (right). Labels on left graph are arbitrary, and the image of each point on the right graph is shown. Colors are used for additional visual aid. Colors on right graph are arbitrary, and points on left graph are colored according to their image under the mapping. Marked points are the 3 extremal corner points on each graph (which correspond to states in which all disks are on top of a single peg).

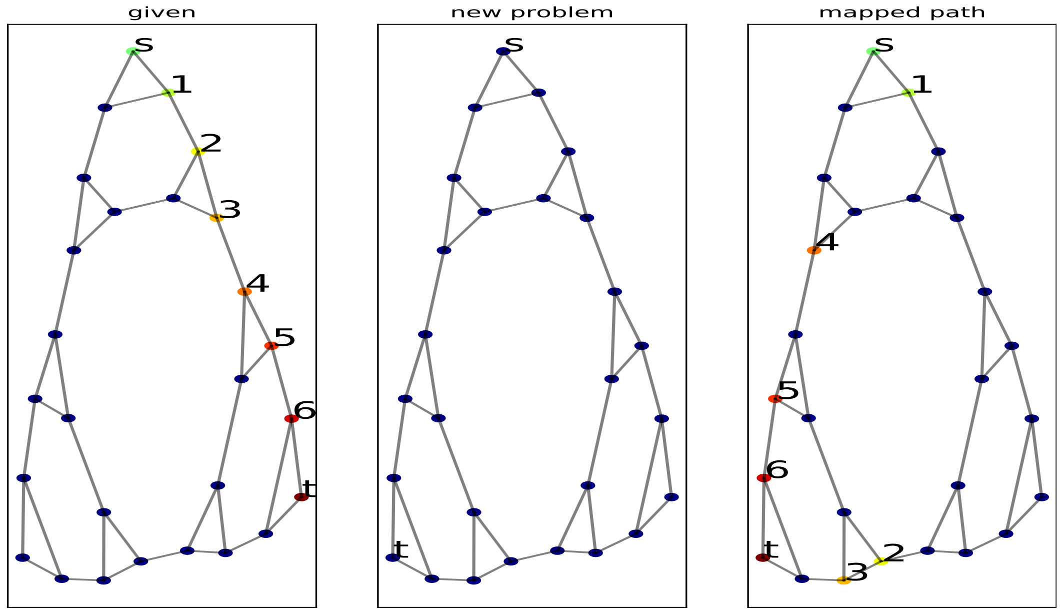

Figure 5: Same as Fig. 1, but with indefinite loss function. Comparison with Fig. 1 shows the L1 loss yields a better matching.

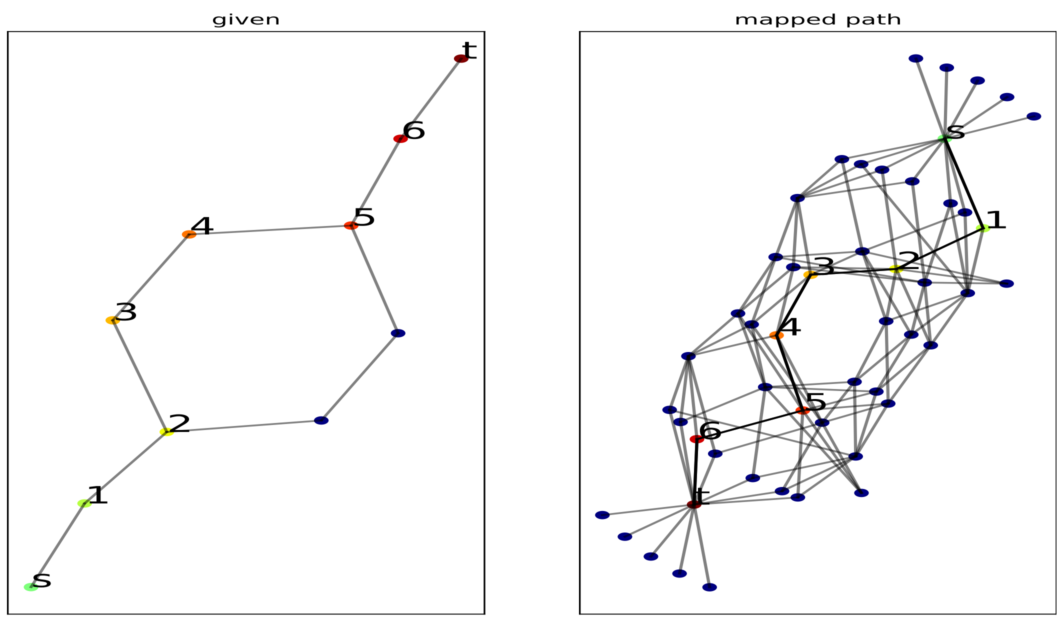

Figure 6: Same as Fig. 2, but with indefinite loss function. Results are comparable to L1 loss.

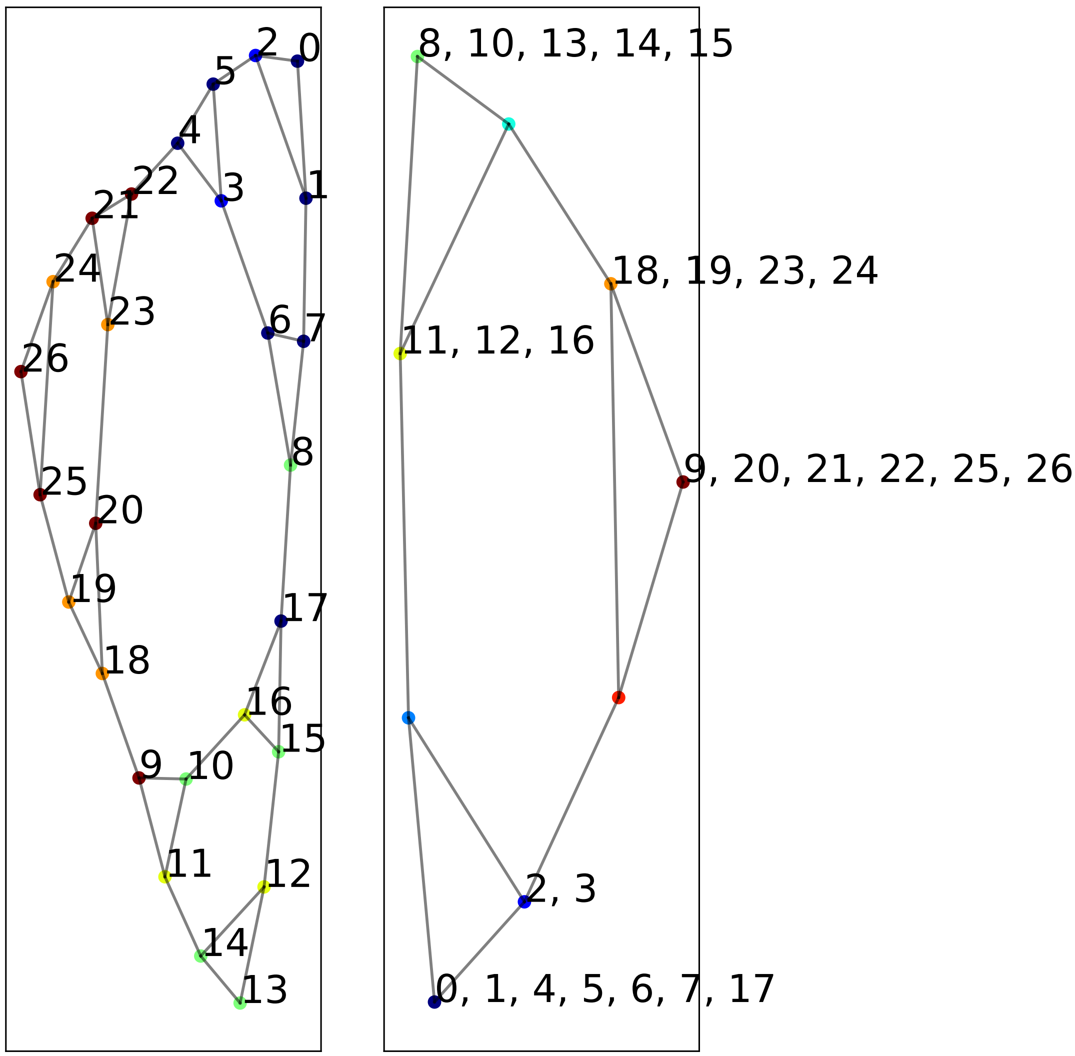

Figure 7: Same as Fig. 4, but with indefinite loss function.Comparison with Fig. 4 shows the L1 loss yields a better matching.

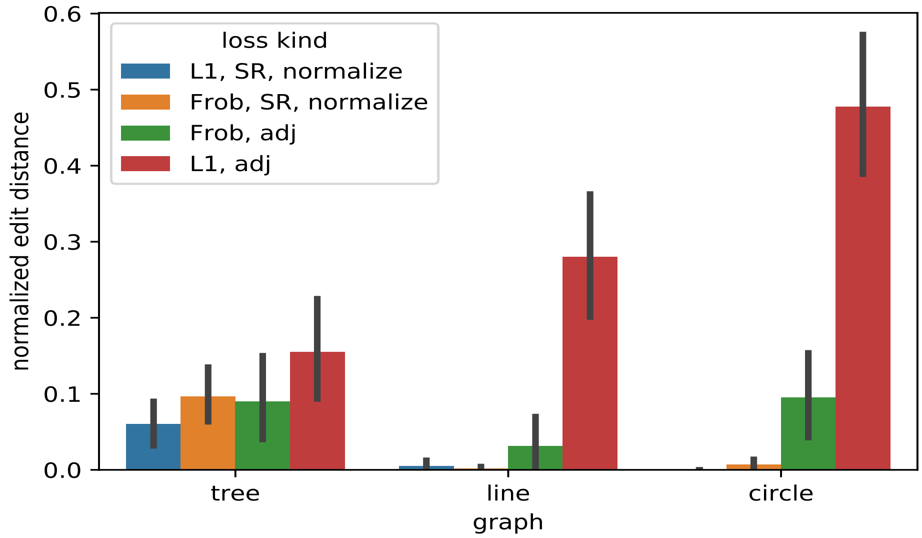

Figure 8: Comparison of quality of generated analogies using different loss functions on different model graphs (lower is better). Blue is our default loss $\overline{L}(\cdot|\mathbf{S}_1, \mathbf{S}_2)$ as described in Section 2. Green is "indefinite loss" of (Lyzinski et al., 2015). Orange is variant of our default loss, replacing L1 with Frobenius norm. Red is the loss from Eq. (2). Our loss (blue) generally outperforms the indefinite loss on these examples.

which encodes whether or not each step in the path goes "against the grain" with respect to the edge directions. The path from C to D_{pred} has a similar sequence. We then define the error of the analogy to be the normalized levenshtein distance between these two sequences. This has the property that the values lie in $[0, 1]$ with 0 indicating a perfect fit. [4]

C.2 Defining the matching problem

Define G_1 to be the induced subgraph of G containing all nodes no more than distance $d_G(A, B)$ from A and G_2 the induced subgraph contains all nodes no more than that distance from C. Let $\mathbf{S}_1$ and $\mathbf{S}_2$ denote the matrices obtained by keeping only the rows and columns of $\mathbf{S}$ corresponding to points in G_1/G_2.

We then solve the marked matching problem $\overline{L}_{\{(A,C)\}}(\mathbf{P}|\mathbf{S}_1, \mathbf{S}_2)$. We also compare this to the indefinite loss function $\|\mathbf{P}\mathbf{A}_2\mathbf{P}^T - \mathbf{A}_1\|_F^2$ from (Lyzinski et al., 2015) and the variant of replacing the L1 norm in ours with the Frobenius: $\|\overline{\mathbf{P}\mathbf{S}_1\mathbf{P}^T} - \overline{\mathbf{S}_2}\|_F^2$.

As described previously, we complete the analogy by taking an argmax of $\mathbf{P}[B]$. Let us denote this point by D_{pred}.

C.3 Results

Comparison of our loss with the indefinite loss is shown in Fig. 8. Our loss $\overline{L}_{(A,C)}(\cdot|\mathbf{S}_1, \mathbf{S}_2)$ achieves highest quality on the tree and circle, and is only slightly worse than the Frobenius variant on the line. The worst performing variant is the $L1$ norm combined with the adjacency matrix.

[4]Note that on these graphs each pair of points has a unique shortest path so there is no ambiguity in this measure. The unique exception is the pair of antipodal points on the circle, however this does not occur in our examples due to the restriction on the distance between the two terms in the analogy.

D Derivation of matrix form for matching problem

Continuing the notation from Sec. 2, we see that

$$N(F) = \Big|\{(v_1, v_2) : (v_1, v_2) \in E_1, (F(v_1), F(v_2)) \notin E_2 \\ \text{or } (v_1, v_2) \notin E_1, (F(v_1), F(v_2)) \in E_2\}\Big| \tag{5}$$

$$= \left|\{(v_1, v_2) \in V_1 \times V_1 : \mathbf{1}_{(v_1,v_2)\in E_1} \neq \mathbf{1}_{(F(v_1),F(v_2))\in E_2}\}\right| \tag{6}$$

$$= \sum_{(v_1,v_2)\in V_1\times V_1} \left|\mathbf{1}_{(v_1,v_2)\in E_1} - \mathbf{1}_{(F(v_1),F(v_2))\in E_2}\right|, \tag{7}$$

where $\mathbf{1}_A$ is equal to 1/0 depending on whether the proposition A is true/false.

Now recall the matrix matrix $\mathbf{Q}_F$ defined by $\mathbf{Q}_F[i,j] = 1$ if $F(i) = j$ and 0 otherwise. We see that

$$(\mathbf{Q}_F \mathbf{A}_2 \mathbf{Q}_F^T)[i,j] = \sum_{k,l} \mathbf{Q}_F[i,k]\mathbf{A}_2[k,l]\mathbf{Q}_F[j,l] \tag{8}$$

$$= \sum_{k,l} \mathbf{1}_{F(i)=k}\mathbf{1}_{F(j)=l}\mathbf{1}_{(k,l)\in E_2} \tag{9}$$

$$= \mathbf{1}_{(F(i),F(j))\in E_2}. \tag{10}$$

Since $\mathbf{A}_1[i,j] = \mathbf{1}_{(v_i,v_j)\in E_1}$, we see that $\|\mathbf{Q}_F \mathbf{A}_2 \mathbf{Q}_F^T - \mathbf{A_1}\|_1 = N(F)$ as was to be shown.

E Solving the continuous problem.

To minimize the continuous objective Eq. (2), let $M_{n,m}$ denote the space of all matrices with n rows and m columns. We parametrize SM_X by the map $Mat_{|V_1|-|X|,|V_2|} \to SM_X$ which takes a softmax across each row and then inserts the clamped rows specified by X in the appropriate order. This allows us to consider L_X as a function on $Mat_{|V_1|-|X|,|V_2|}$; because this is a linear space, we can minimize the objective using standard gradient descent. The variants in the “Variants” subsection are treated in the same way. We use 20 random initializations for each problem.

F Description of Farmer/Goat and Missionaries/Cannibals

As in (Holyoak & Thagard, 1989), the Farmer/Goat problem consists of a farmer, a wolf, a goat, and a cabbage. All parties start on the same shore of the river, and the objective is to transport them all to the opposite shore. The boat can only hold one item in addition to the farmer. The wolf cannot be left alone with the goat, nor can the goat with the cabbage.

The Missionaries/Cannibals problem consists of 3 missionaries and 2 cannibals which must all cross the river. The boat can hold at most two parties, and the missionaries can never be outnumbered the cannibals on either shore. Additionally, it is disallowed to make a trip across the river with an empty boat.