Evolutionary Graph Compression and Diffusion Methods for City Discovery in Role Playing Games

Joseph Alexander Brown
AI in Games Development Lab
Innopolis University
Innopolis, Russia
j.brown@innopolis.ru

Daniel Ashlock
Mathematics & Statistics
University of Guelph
Guelph, ON, Canada
dashlock@uoguelph.ca

Sheridan Houghten
Computer Science
Brock University
St. Catharines, ON, Canada
shoughten@brocku.ca

Angelo Romualdo
Computer Science
Brock University
St. Catharines, ON, Canada
angelo.m.romualdo@gmail.com

***Abstract*—Cities, while exciting in their visualization and permitting several layouts, do not take into account the placement of crucial characters which might be part of the narrative. Narrative graphs, a connected graph of all potential and existing relations within a game, can enable an ability to find a Non-player Character (NPC) who is likely to live nearby, under the assumption that those who interact most frequently are also close in distance. We examine the use of an evolutionary graph compression method and a method using simulated diffusion to cluster features based on relational information about players to generate relationally intimate groups. This clustering can be used to generate information about the game world and cities to inform PCG as to how the connectivity of these areas is, and should be, arranged. The algorithms are validated as being human competitive.**

***Index Terms*—Procedural content generation, narrative graphs, diffusion characters, compression, clustering**

I. Introduction

Procedural Content Generation (PCG), see [30] for an overview, provides developers with generators based on algorithmic means to potentially speed up the developmental process and allow for replayability. While several generators exist for the creation of cities [29] and building interiors [12], [18], [25] and exteriors [15], few take into account the narrative factors of the relationships of non-player characters (NPCs) within them and as a result, are unable to make discoveries.

As more procedural content development tools for NPCs are created, and as they focus more on narrative factors, the discovery of natural relationships among players in the placement of NPCs becomes more relevant. The study [10] examines a number of rules for the relations NPCs should have. In [11] a number of NPC communications were examined for both small world graphs and with real game graphs. In [13], this graph framework was applied to emotional outcomes due to interactions between NPCs.

A. Main Goal

The goal of this work is to use narrative requirements, social structures, and networks as a basis for a generative requirement, namely the placement of cities and people. In doing so, this frees designers to concentrate solely on the narrative aspects, with the proposed methods taking care of the generation of appropriate links and relationships between NPCs and their placement within cities.

Narrative aspects are represented in a player relation graph, which we propose to mine in two ways in order to allow for the automatic discovery of logical cities based on NPC relationships. The first is *compression* of the graph as a precursor to a city or level generation. The second uses *simulated gas diffusion*, termed diffusion characters, to inject the graph into Euclidean space for visualization and analysis. We examine the performance of the two methods when applied to the problem of extracting information from player relation graphs.

Discovering the relational links between NPCs, created from the narrative communities, naturally forms communities in the player space, which then map smoothly to the formal communities created by PCG. The proposed methods permit the discovery of natural cities arising from social relationships.

We apply our methods to a number of graphs developed from the NPCs of four actual games: *Fallout 4* [27], *New Vegas* [7], *The Elder Scrolls IV: Oblivion* [6], and *The Elder Scrolls V: Skyrim* [8]. These were selected due to the similarities in their design patterns allowing for the creation of graph networks of conversations between characters, and they are representative of this class of games. The purpose of examining these existing graphs is to act as a proof of concept, namely, to demonstrate that our methods are able to provide a human competitive approach to the problem by comparing their output against a ground truth created by a designer. The methods are given only the narrative context of the relationships between NPCs, from which they are then expected to generate a potential city clustering. The developer can thus focus on the narrative aspects of the game.

The remainder of this paper is organized as follows. Section II provides background information on graphs, compression and diffusion characters. Section III describes the process followed to extract information from the existing games, to be used as input and validation for our methods. Section IV provides detail on the two proposed methods, with the corresponding results in Section V. Sections VI and VII detail conclusions and discuss possible future work.

II. Background

A *graph* has a set of *nodes* (or *vertices*) and a set of *edges*, where each edge connects two nodes; when there is an edge between two nodes then those nodes are *neighbours* of one another. A *path* between two nodes is a sequence of edges that connect the nodes; the *distance* between two nodes is the length of the shortest path between the nodes.

This study tests two methods, *graph compression* and *diffusion character analysis*, on graphs in which vertices represent NPCs and edges represent interactions between them. These are standard simple, undirected graphs. Only a very basic knowledge of graph theory is required: the graph stores the interaction structure, and the analysis techniques attempt to recover the communities within the game, where the NPCs resided.

A. Graph Compression

There are many different forms of graph compression schemes [5]. Graph compression is studied for various reasons. Studies on graph compression may be organized into three main groups: the first studies the structure of graphs, the second concerns graph partitioning, and the last group solves challenges in large graph mining.

1) Graph structure: Most real-world graphs exhibit interesting properties which distinguish them from random graphs such as power-law degree distributions [22]. However, processing large graphs and representing them in a meaningful form is very expensive. Graph compression allows the scalable processing of such graphs.

The algorithms in [17] and [23] compress graphs by removal of hub nodes. After the removal of the hub nodes, many nodes "shatter" and become disconnected. The authors repeated the process of removal until satisfied, noting that this is most effective on graphs that contain multiple hubs recursively connected to larger hubs.

In the current study, we are working with graphs drawn from game environments that are known *a priori*. We consider algorithms for detecting community structure and use them to attempt to recover the known community structure from the game environment in a number of commercially available games. By doing so, we validate the algorithm as being human competitive to a game designer.

2) Graph partition and compression: One of the most extensively studied topics in network analysis is community detection or clustering. The main task is to find a set of nodes in a graph that is homogeneous and group them. Clustering naturally implies that the resulting graph would have clusters of nodes, and therefore, we can partition nodes into groups. Once the graph is partitioned, we can compress each group with a representative node. One of the most successful methods to partition graphs is *multilevel graph partitioning*, which leverages the structural information.

In social networks, the graph may be partitioned in terms of node attributes. For instance, in twitter networks, we may partition the graph in terms of tweet topics [28].

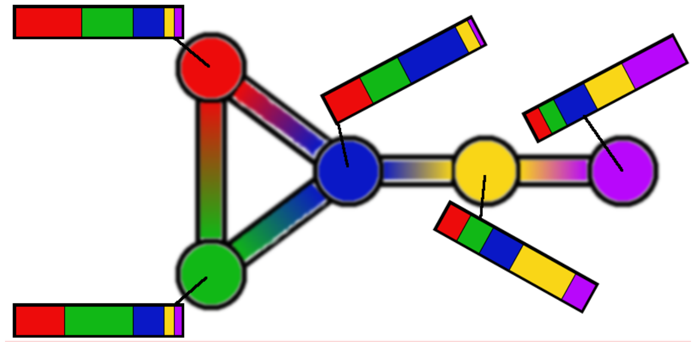

Fig. 1. Shown is a five-vertex graph with five types of gas, represented by different colours. The coloured bars associated with each vertex represent the gas distribution at the vertex. Converting the relative gas abundance into fractions generates the coordinates of the point associated with the vertex by the diffusion character.

3) Large graph mining: As the sizes of graphs have grown, many adaptive techniques to process such enormous datasets have been developed, such as machine learning algorithms and MapReduce. However, to be able to cope with exponentially growing graph datasets, new parallel abstractions like Google's Pregel [24] have been introduced. Pregel enables the processing of large graphs in a scalable manner by vertex-centric computations. Depending on the types of graph datasets, many different properties of nodes can be exploited to compress the graph with Pregel, such as similarity and locality. For example, [14] used link reciprocity to compress social networks.

B. Diffusion Characters on Graphs

Diffusion characters [4] are a technique for injecting the vertices of a graph into Euclidean space in a number of dimensions equal to the number of vertices in a graph. This permits analysis of the graph and the relationships it encodes by means that operate on feature vectors with a fixed number of dimensions. Each vertex in a graph has a diffusion character centred on it, and the coordinates of this diffusion character form the coordinates of the point in Euclidean space representing it.

The diffusion character of a vertex is computed by permitting a gas, perpetually renewed at the vertex, to diffuse across the graph. One unit of gas is added, in each time step, to the central vertex and the amount of the gas at each vertex is then divided equally among the vertex and its neighbours. After the gas has moved to neighbour vertices, the amount of gas at each vertex is then multiplied by decay parameter α, $0 < \alpha < 1$, set to $\alpha = 0.99$ in this study. This decay represents the absorption of the gas at each vertex. There is a separate diffusion character, with a different type of gas, centred at each vertex. A graph with five vertices is shown with its gas diffusion in Figure 1. The amount of the gas associated with the diffusion character centred at vertex i present at vertex j, once the system has come to equilibrium, is a measure of the connectivity from j to i. The measure is not a metric (it does not obey the three metric space axioms) and is not symmetric. This latter fact agrees with the behaviour of random walks.

A random walk starting at a high degree vertex has more potential destinations than one starting at a low degree vertex, and so the probability of walking from a low degree vertex to a high degree vertex is asymmetric. The conservation law we impose on gas diffusion neatly captures this fact.

The absorption, encoded by the decay parameter α, ensures that the system comes to equilibrium. In a theorem from [21] it is demonstrated that the system of diffusion character equations can be made well conditioned by choice of a parameter and so may safely be computed by the solution of a linear system or by simulation (when the number of vertices in the graph makes a direct solution by matrix inversion impractical). The mathematics of diffusion characters is explained in substantial detail in [21]. They have been used in network analysis in several past studies [1], [2] [4].

The concentration of each type of gas drops off exponentially with the distance from the vertex where that gas originates. For that reason, the values of the diffusion characters are negative-log transformed to create the coordinates of points in Euclidean space. The standard Euclidean metric on those points then places an alternate distance measure on the vertices representing NPCs in a social graph.

III. Graphs from Games

A. Process

The process assumes the existence of a narrative graph of relations between a set of characters as a basis for the generation. This could be directed about interactions in a quest line or based on interpersonal relationships existing in the narrative of the game. The presence of these relationships once found in the game can be placed into a graph structure, in which nodes are the entities in the game, and the edges represent the connections of positive relationships between them.

B. Test Graphs

We evaluate our system using a series of Bethesda games in both the Elder Scrolls and Fallout Universes. We created NPC connection graphs for each of the four games. Spatial information (i.e. cities) is also known for each of these games. We evaluate the ability of the system to discover these cities based on the narrative information (NPC interaction graphs) alone. If successful, this indicates that narrative information can be used to generate spatial information; as stated in Section I-A, this then allows the developer to concentrate solely on narrative aspects while using the system to generate the spatial aspects.

For all of the four games, NPCs were considered to have a relationship if they at any time entered the same loading sector (known as a *CellName*) in the game. Note this means there are a small number of narrative-driven links that may not happen in a playthrough of the games but that have the potential of occurring, which are treated as having an equally strong link as two people who are constantly in communication who live in the same village. We have removed several NPCs that do not have a home CellName. Where possible, we have also included downloadable content (DLC) to obtain the data. Also, we have limited the study to assigning characters to major cities in the game and have removed factional sectors (such as the Prydwen and Institute in *Fallout 4* or Caesar's legion in *Fallout: New Vegas*) in which there are no communication links outside of the faction's base, as it would be an obvious design choice for the designer to just place these functionaries in the fortress-monastery of the Faction, and we are interested in the overworld being created. They also form a small non-connected component of the graph – making their detection trivial as compared to a factional base like Acadia in *Fallout 4*, which does have external communicative links.

1) Fallout: New Vegas: Fallout New Vegas [7] contains a graph of 151 NPCs. There are eight large cities in the area: Boulder City, Freeside, Goodsprings, New Vegas, Hopeville, Jacobstown, Red Rock Canyon, and Westside.

2) Fallout 4: Fallout 4 [27] contains a graph of 152 NPCs. There are eight large cities in the area: Acadia, Bunker Hill, Covenant, Diamond City, Far Harbor, Goodneighbor, The Nucleus, and Vault 81. See Figure 2.

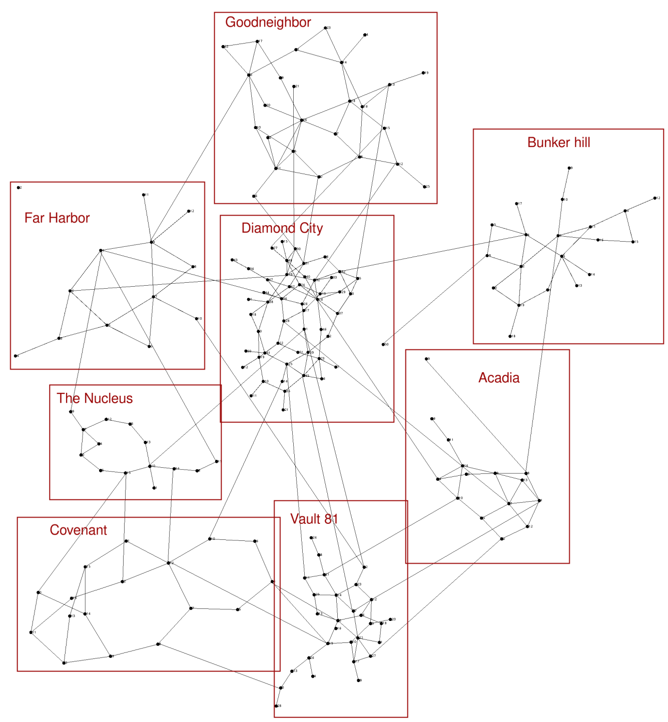

Fig. 2. Fallout 4 Graph with cities denoted by a Human

3) Oblivion: Elder Scrolls: Oblivion [6] contains a graph of 659 NPCs. There are nine large cities in the area: The Imperial City, Anvil, Bravil, Bruma, Cheydinhal, Chorrol, Kvatch, Leyawin, and Skingrad. See Figure 3.

4) Skyrim: Skyrim [8] contains a graph of 612 NPCs. There are nine large cities in the area: Markarsh, Rifter, Solitude, Windhelm, Dawnstar, Falkreath, Morthal, and Winterhold.

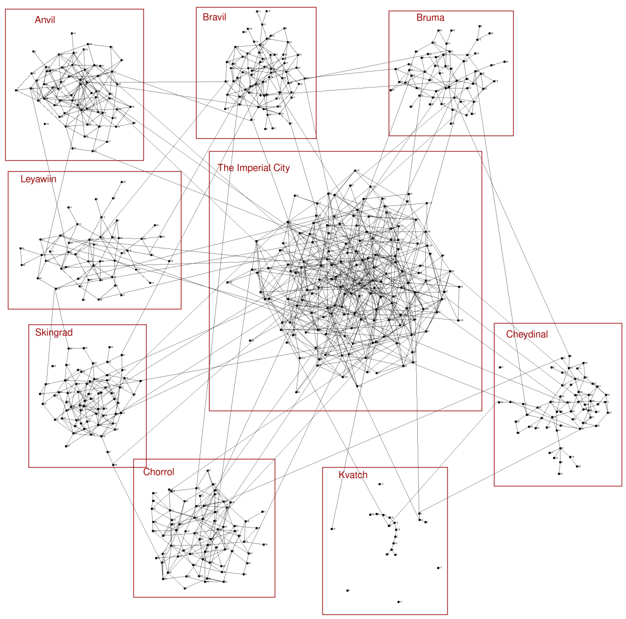

Fig. 3. Oblivion Graph with cities denoted by a Human

IV. Methods

A. Compression

In the current study, a hierarchical approach is used. In this approach, a graph is compressed by the merging of sets of nodes into *supernodes* and sets of edges into *superedges*. Examples of such an approach include [26] and [31]. The current study employs a genetic algorithm (GA) to select which nodes to merge according to a fitness function that measures the distortion created by the merges. Nodes that are merged together are considered to have some sort of close relationship or similarity. From the perspective of the game graphs, therefore, supernodes are essentially clusters of nodes (NPCs) that are closely related. By finding these clusters, we have evidence for where a generative placement of a city should be within the game world.

The representation and methodology have evolved over the course of a number of previous studies (see, e.g. [9], [32], [16], [20]) and used successfully to address the compression of biological data. A brief description of the representation follows, and we refer the reader to these earlier studies for further details.

1) Representation: The compression is accomplished by a sequence of merges, with each step merging two nodes. The chromosome represents the sequence of merges as a pair of one-dimensional arrays, with matching elements representing the two nodes in a given merge. The index of the first node in merge i is `root[i]` and the index of the second node in merge i is `(root[i] + offset[i])` $mod\ N_o$, where N_o is the number of nodes in the original graph.

root	4	7	75	20	17
offset	55	88	57	30	93

Fig. 4. Example Chromosome

root	4	**8**	75	20	17
offset	55	**11**	57	30	93

Fig. 5. Result of Mutation

To compress a graph with N_o nodes by a compression ratio of C requires a sequence of $C * N_o$ merges to produce a graph with $N_c = N_o - C * N_o$ nodes. The desired compression ratio is given as an input parameter. A graph with 100 nodes to be compressed by 5% requires 5 merges, so the chromosome consists of a pair of arrays of length 5. In the example chromosome shown in Fig. 4, the following nodes are merged: node 4 with node $(4 + 55)$ *mod* $100 = 59$, node 7 with node $(7 + 88)$ *mod* $100 = 95$, node 75 with node $(75 + 57)$ *mod* $100 = 32$, node 20 with node $(20 + 30)$ *mod* $100 = 50$ and node 17 with node $(17 + 93)$ *mod* $100 = 10$.

2) Local Merges: As in [20], all merges must be *local*. In a local merge, the nodes are within a specified distance of each other. From the perspective of the game graphs considered in the current study, this restriction should help to prevent merges between nodes (NPCs) in different cities, thereby helping to ensure that nodes are only merged together if they are closely related from a game perspective.

3) Initial Population: The first node of each merge is chosen randomly, and then a breadth-first search is performed from that node to find all other nodes within the specified distance; one is then chosen at random as the second node.

4) Selection: Tournament selection is used: for each parent, k chromosomes are chosen at random from the population and evaluated, with the best selected for reproduction. These parents are then subjected to crossover and mutation based on the settings to create two children. The process is repeated to create all chromosomes for the next generation.

5) Mutation: A single-point mutation is used, having the effect of changing a single merge. The mutation point j is chosen randomly. First, `root[j]` is changed to a random value between 0 and $N_o - 1$. Next, to ensure that the merge will be local, the other node is chosen randomly from among all other nodes within the specified distance of `root[j]`; from among these, one is chosen at random and `offset[j]` is set accordingly. Fig. 5 shows the result of a single-point mutation in the 2nd entry of the chromosome from Fig. 4.

6) Crossover: Two-point crossover is used. The first crossover point is a random value between 1 and $N_o - N_c$, the size of the chromosome, and the second point is a random value between the first crossover point and $N_o - N_c$. Fig. 7 shows the result of crossover on the pair of chromosomes from Fig. 6, with the second and third entries both being exchanged between the two chromosomes.

7) Fitness: The fitness function measures the distortion created by the sequence of merges, where we consider distortion to be the creation of additional edges that did not exist in the

rootA	4	7	75	20	17
offsetA	55	88	57	30	93

rootB	9	22	46	20	85
offsetB	11	35	49	14	1

Fig. 6. Before Crossover

rootA	**4**	22	46	**20**	**17**
offsetA	**55**	35	49	**30**	**93**

rootB	9	**7**	**75**	20	85
offsetB	11	**88**	**57**	14	1

Fig. 7. After Crossover

original graph. In merging two nodes into a *supernode*, upon decompression, this supernode becomes a clique, which may create additional edges in comparison to the original graph. The fitness, which should be minimized, counts the number of such additional edges created. For further detail, we refer the reader to [20].

B. Diffusion Character Methods

The diffusion characters for the four graphs in this study are computed by direct simulation of the gas diffusion, running the diffusion for a long time, over several time steps equal to three times the number of vertices, to ensure that the level of each gas at each vertex is close to equilibrium. The vertices are then injected into $\mathbb{R}^n$ where n is the number of types of gas (equivalently, number of vertices in the graph), taking the negative log transform of the gas levels for each gas at each vertex. This creates a cloud of points that can be clustered and visualized.

Non-linear projection (NLP) [3] is a method for projecting a cloud of points in n dimensions into a lower-dimensional space for visualization. NLP is a form of multidimensional scaling performed with evolutionary computation. A survey of techniques for multidimensional scaling is [19], which gives many fast heuristics and variations on multidimensional scaling. The goal of nonlinear projection is to provide a relatively faithful projection of points from a high-dimensional space into a two-dimensional space that distorts the inter-point distances as little as possible.

In order to perform NLP, we evolve the coordinates of a collection of points in two dimensions, one for each point in the higher dimensional space, and maximize the Pearson correlation coefficient, given in Equation 1, of the original distances with distances of points in two dimensions. We restrict the two-dimensional point cloud to the unit square; Pearson correlation of distance matrices is invariant under translation, rotation, reflection across a line, and scaling of the projected points. Use of Pearson correlation as a fitness function permits the evolutionary algorithm to solve the problem of relative distance without worrying about other quantities, yielding a faithful if scaleless visualization.

The Pearson correlation coefficient is given by:

$$cor = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{(n-1)s_x s_y} \tag{1}$$

Where for $z \in \{x, y\}$, $\overline{z}$ denotes the sample mean and s_z denotes the sample standard deviation. This is applied to the entries of the distance matrices for the high-dimensional points and the projected points.

V. Results

A. Compression Results

We compressed each of the four-game graphs using compression ratios of 10%, 25% and 40%; for each of these, the maximum allowed distance between merged nodes was set to 3 and 5, thus producing a total of six trials for each of the graphs. For all test cases, we use the GA settings as specified in Table I; these were determined empirically.

In the compression, a merge is deemed to be *correct* if it joined two NPCs from the same city. Table II shows the fraction of correct merges, for the run with the highest fitness for each of the four graphs and six trials. The results clearly show the value of using an appropriate distance to force all merges to be local. In all but one case, the percentage of correct merges was higher for distance 3 than for distance 5. Even at 40% compression, with distance 3, the percentage of correct merges was 100% for New Vegas, 98.3% for Fallout4, 99.6% for Skyrim and 93.9% for Oblivion, which is the largest graph.

We remind the reader that the goal of this study is to demonstrate the potential of such methods in creating a suitable spatial structure based solely on narrative information. The above results show that the described compression methodology will reliably compress a graph based on NPC interactions, such that when two nodes (NPCs) are merged together, they have a very high likelihood of belonging to the same city when the spatial structure is known *a priori*. In the future, the designer can simply provide narrative information, i.e. a set of known NPC interactions that form a narrative graph, and use the compression methodology to *generate* the spatial structure. Specifically, if a set of nodes (NPCs) are merged together by compression of the narrative graph, then the developer can have high confidence that these NPCs should be in the same city and thus create cities based on that information. This allows for complex but sensible spatial structures to be easily built from even very large sets of NPC interactions.

B. Diffusion Character Results

Both k-means clustering and NLP were performed to see how well the non-player characters, represented as graph vertices, could be classified as belonging to their known city. The k-means results were terrible while the NLP visualizations yielded promising results that also explain the problems with the k-means results. The NLP projections showing the distributions of the NPCs are shown in Figure 8. These results show good to excellent separation of the agents into the correct community groups.

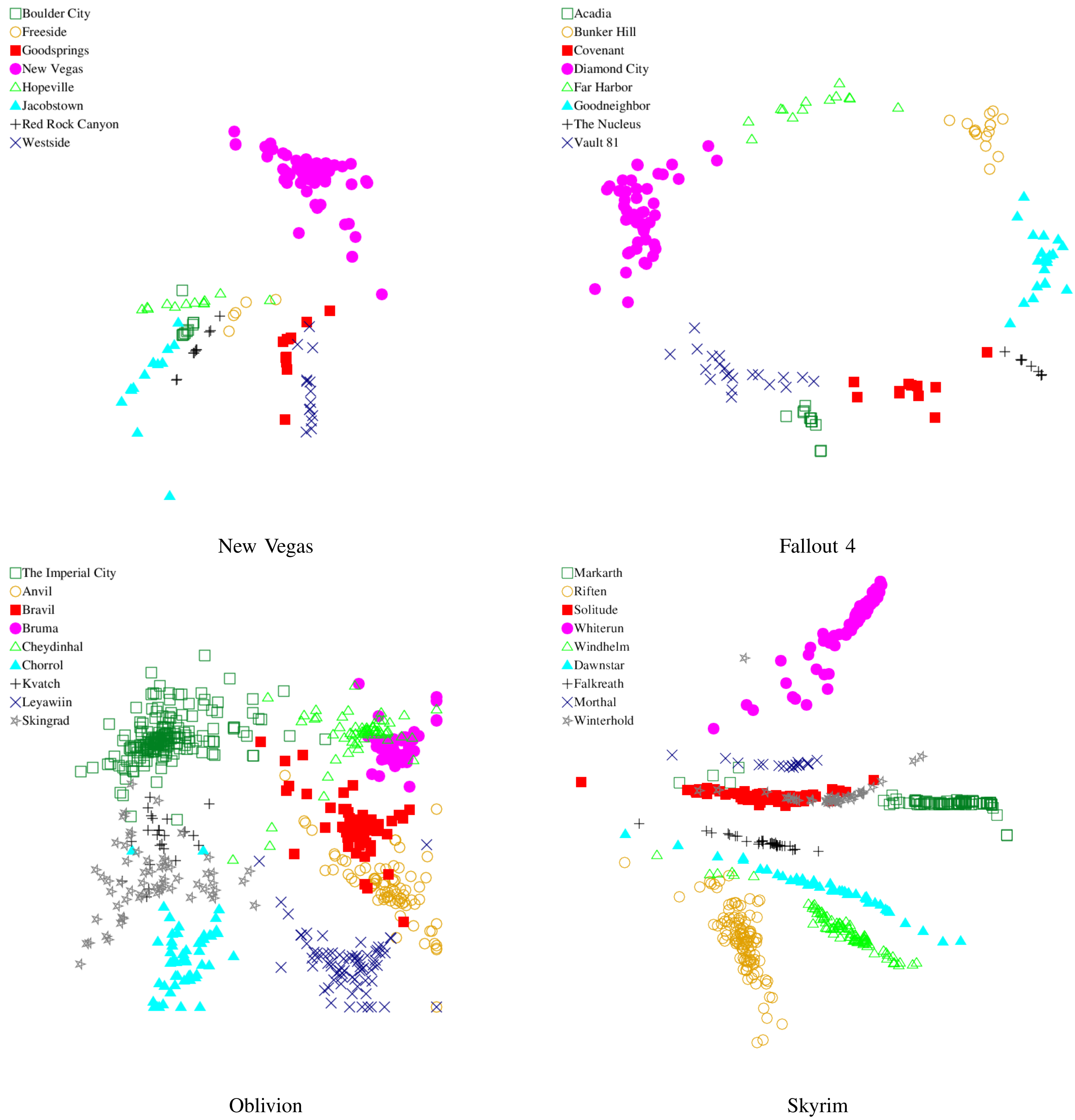

Fig. 8. Non-linear projections of the log-transformed diffusion characters for the four game communities used in this study.

All four clouds of points generated as log-transformed diffusion characters were run through k-means clustering with k chosen to be the number of communities, a known number of clusters, and a rare luxury in using k-means. The clustering always grouped distinct communities as single clusters. First of all, it is clear from the results in Figure 8 that some of the communities have strong interpersonal linkage and so are not unreasonably associated. A more significant issue is that the communities detected seldom form a compact shape. The k-means algorithm must find compact clusters, or to be more precise, sets of points that fit within a convex polytope, with one such polytope for each cluster. The long thin clusters that dominate the Skyrim results, and which appear in all the visualizations, are likely to either be split (if we had used more clusters) or fuse with a nearby community chosen so that the union of the two communities makes a compact shape. Many

TABLE I
EXPERIMENTAL PARAMETERS FOR THE COMPRESSION GA

Parameter	Value
Mutation Rate	10%
Crossover Rate	90%
Generations	500
Population Size	100
Tournament Size	5
Number of Elites	1
Number of Runs	10

TABLE II
PERCENTAGE OF MERGES THAT INVOLVED TWO RESIDENTS OF THE SAME CITY. FOR EACH OF THE GRAPHS, SIX TRIALS WERE PERFORMED WITH DIFFERENT COMPRESSION AND DISTANCE PARAMETERS.

New Vegas	**Comp**.	**Dist.**	**Fallout 4**	**Comp.**	**Dist.**
100%	10%	3	93.3%	10%	3
100%	10%	5	100%	10%	5
100%	25%	3	94.7%	25%	3
100%	25%	5	94.7%	25%	5
100%	40%	3	98.3%	40%	3
93.3%	40%	5	91.6%	40%	5
Oblivion	**Comp.**	**Dist.**	**Skyrim**	**Comp.**	**Dist.**
100%	10%	3	100%	10%	3
90.8%	10%	5	96.7%	10%	5
98.2%	25%	3	100%	25%	3
85.4%	25%	5	95.4%	25%	5
93.9%	40%	3	99.6%	40%	3
76.8%	40%	5	91.0%	40%	5

other clustering techniques could be tried, and the NLP results will be instructive in the selection of an effective one.

Historical connections between some of the regions due to their narratives are also seen. Looking at the cities of Winterhold and Solitude in *Skyrim*, the capital was once in Winterhold and then moved to Solitude, leading to a rivalry between the cities which is seen in the movement of some NPCs between the locations. Similarly, Acadia is tightly connected in *Fallout 4* being one of the locations in the Far Harbour DLC in which the Synths have shelter away from the number of factions.

As with the compression results, the diffusion character results detected either cohesive cities or, for New Vegas, well-connected city cores, using only narrative information. This shows that diffusion character results, based solely on narrative information, can be used to gather together players that would form natural cities within the game world. Both methods thus achieve the stated goals of detecting cities from player interactions.

VI. CONCLUSIONS

The goal of this work was to use the narrative requirements, social structures and networks, as a basis for a generative requirement, the placement of cities and people. We examine the performance of two methods, graph compression and simulated gas diffusion, when applied to the problem of extracting information from player relation graphs.

In order to demonstrate the methods, we used four communication networks from commercially developed games as base truth information. These graphs show highly connected cliques with weak connections between them centred about the cities. Both techniques used discover the existing cities with a high probability of success. Therefore, we would see this as evidence of the techniques' suitability for games during the development phase, and for being part of a PCG pipeline that respects the narrative structures of the system created by NPC interactions.

We do not view either of these techniques as producing inherently better outcomes from the perspective of the definition of the placement of cities Hence, we present both as viable methods of design, and depending on other factors in the development and the data available to the games designer, either could be selected as a reasonable approach this problem. In order to see a clear use-case for one method over the other, we would need to examine other factors in actual development, e.g. speed, offline/online development, narrative requirements, etc.

VII. FUTURE WORK

There are other narrative requirements examined, such as quest lines providing information to the generational process. The clustering using narrative relationships allows new city layouts to be produced. It can also act as a tool for designers in order to verify assumptions about the quest lines. If several highly connected NPCs are placed in a group by the clustering but are supposed by the designers to live about the world, then they are violating the assumption that close relationships are formed between physically close characters. In such a case, this can be addressed either by adding more relationships to force the relations to be meet with the assumption, or this implies a violation of the assumption of proximity. Violations should have a narrative rationale, which is assumed to be a city-like grouping, for example, a hidden spy network that keeps a member of their order in each city and clandestinely communicates. Further, the method of examination of proposed cities can also act as a check for excessive backtracking in quest lines.

In order to examine the stability of the approaches, it would be best to introduce several noise-introducing factors. If developers make minor changes to the narrative relationships between characters, this should not make significant changes to the expected clustering of the graph of the cities. Also, the links in our game graphs do not take into account the factional natures, the type of communication, or the strength of the bonds between the NPCs. If we had information about the types of links, we could better split graphs based upon other factors such as occupation or factional biases and even give back narrative actions to the next generator. For example, a trader is likely to be on the edge of two cities as he moves gossip and information between them, whereas it would be unlikely for two enemy factions to be clustered in the same city without a local conflict. These concerns are, of course, highly dependent upon the problem case.

Datasets

The datasets for the graphs are publicly available at: http://www.cosc.brocku.ca/~houghten/gamegraphs.html

Acknowledgements

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] D. Ashlock and C. Lee. Characterization of extremal epidemic networks with diffusion characters. In *Proceedings of the 2008 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 264–271, 2008.

[2] D. Ashlock and C. Lee. Using diffusion characters for the taxonomy of self-organizing social networks. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 60–67, 2009.

[3] D. Ashlock and A. McEachern. Evolutionary nonlinear projection. *IEEE Transactions on Evolutionary Computation*, 6(19):857–869, 2015.

[4] D. Ashlock, E. Shiller, and C. Lee. Comparison of evolved epidemic networks with diffusion characters. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 781–788, 2011.

[5] M. Besta and T. Hoefler. Survey and taxonomy of lossless graph compression and space-efficient graph representations. *arXiv preprint arXiv:1806.01799*, 2018.

[6] Bethesda Game Studios. *The Elder Scrolls IV: Oblivion*. Bethesda Softworks, 2006.

[7] Bethesda Game Studios. *Fallout : New Vegas*. Bethesda Softworks, 2010.

[8] Bethesda Game Studios. *The Elder Scrolls V: Skyrim*. Bethesda Softworks, 2011.

[9] J. A. Brown, S. Houghten, T. K. Collins, and Q. Qu. Evolving graph compression using similarity measures for bioinformatics applications. In *2016 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–6, Oct 2016.

[10] J. A. Brown and Q. Qu. Systems for player reputation with npc agents. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 546–547, Aug 2015.

[11] Joseph Brown, Jooyoung Lee, and Niktia Kraev. Reputation systems for non-player character interactions based on player actions. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2017.

[12] Joseph Brown, Bulat Lutfullin, Pavel Oreshin, and Ilya Pyatkin. Levels for hotline miami 2: Wrong number using procedural content generations. *Computers*, 7(2):22, Apr 2018.

[13] X. Caddle, C. Gittens, and M. Katchabaw. A psychometric detection system to create dynamic psychosocial relationships between non-player characters. In *2018 IEEE Games, Entertainment, Media Conference (GEM)*, pages 256–262, Aug 2018.

[14] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, Michael Mitzenmacher, Alessandro Panconesi, and Prabhakar Raghavan. On compressing social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 219–228, New York, NY, USA, 2009. ACM.

[15] C. Coia and B.J. Ross. Automatic evolution of conceptual building architectures. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, pages 1140–1147, 2011.

[16] T.K. Collins, A. Zakirov, J.A. Brown, and S. Houghten. Single-objective and multi-objective genetic algorithms for compression of biological networks. In *2017 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8, 2017.

[17] Christos Faloutsos and U. Kang. Beyond 'caveman communities': Hubs and spokes for graph compression and mining. *2011 IEEE 11th International Conference on Data Mining (ICDM 2011)*, 00:300–309, 2011.

[18] R. W. J. Flack and B.J. Ross. Evolution of architectural floor plans. In Cecilia Chio, Anthony Brabazon, Gianni A. Caro, Rolf Drechsler, Muddassar Farooq, Jörn Grahl, Gary Greenfield, Christian Prins, Juan Romero, Giovanni Squillero, Ernesto Tarantino, Andrea G.B. Tettamanzi, Neil Urquhart, and A.Şima Uyar, editors, *Applications of Evolutionary Computation*, volume 6625 of *Lecture Notes in Computer Science*, pages 313–322. Springer Berlin Heidelberg, 2011.

[19] P. J. F. Groenen and I. Borg. *Modern Multidimensional Scaling: Theory and Applications*. Springer-Verlag, Secaucus, NJ, 2005.

[20] S.K. Houghten, A. Romualdo, T.K. Collins, and J.A. Brown. Compression of biological networks using a genetic algorithm with localized merge. In *2019 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8, 2019.

[21] C. Lee and D. Ashlock. Diffusion characters: Breaking the spectral barrier. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering 2008*, pages 847–850, 2008.

[22] Jooyoung Lee and Jae C Oh. Estimating the degrees of neighboring nodes in online social networks. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 42–56. Springer International Publishing, 2014.

[23] Yongsub Lim, U. Kang, and Christos Faloutsos. Slashburn: Graph compression and mining beyond caveman communities. *IEEE Trans. Knowl. Data Eng.*, 26(12):3077–3089, 2014.

[24] Grzegorz Malewicz, Matthew H. Austern, Aart J.C Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: A system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 135–146, New York, NY, USA, 2010. ACM.

[25] Paul Merrell, Eric Schkufza, and Vladlen Koltun. Computer-generated residential building layouts. In *ACM SIGGRAPH Asia 2010 papers*, SIGGRAPH ASIA '10, pages 181:1–181:12, New York, NY, USA, 2010. ACM.

[26] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. Graph Summarization with Bounded Error. In *SIGMOD*, pages 419–432, 2008.

[27] Obsidian Entertainment. *Fallout 4*. Bethesda Softworks, 2015.

[28] Qiang Qu, Cen Chen, Christian S. Jensen, and Anders Skovsgaard. Space-time aware behavioral topic modeling for microblog posts. *IEEE Data Eng. Bull.*, 38(2):58–67, 2015.

[29] Christoph Salge, Michael Cerny Green, Rodgrigo Canaan, and Julian Togelius. Generative design in minecraft (gdmc): Settlement generation competition. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, FDG '18, pages 49:1–49:10, New York, NY, USA, 2018. ACM.

[30] Noor Shaker, Julian Togelius, and Mark J. Nelson. *Procedural Content Generation in Games*. Springer, Sweden, 2016.

[31] Hannu Toivonen, Fang Zhou, Aleksi Hartikainen, and Atte Hinkka. Compression of weighted graphs. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 965–973, 2011.

[32] A.N. Zakirov and J.A. Brown. NSGA-II for biological graph compression. *Advanced Studies in Biology*, 9(1):1–7, 2017.