Energetic Path Finding across Massive Terrain Data

Andrew Tsui and Zoë Wood

Abstract. Throughout history, the primary means of transportation for humans has been on foot. We present a software tool which can help visualize and predict where historical trails might lie through the use of a human-centered cost metric, with an emphasis on the ability to generate paths which traverse several thousand kilometers. To accomplish this, various graph simplification and path approximation algorithms are explored. We show that it is possible to restrict the search space for a path finding algorithm while not sacrificing accuracy. Combined with a multi-threaded variant of Dijkstra's shortest path algorithm, we present a tool capable of computing a path of least caloric cost across the contiguous US, a dataset containing over 19 billion datapoints, in under three hours on a 2.5 Ghz dual core processor. The potential archaeological and historical applications are demonstrated on several examples.

1 Introduction

Anthropologists, archaeologists, and historians have spent a great deal of time uncovering the routes taken by ancient travelers on trade-routes with long-distance neighbors, or as they foraged for food around their camp [1]. As a general rule of thumb, humans are often quite proficient in finding the most efficient path of travel, including those paths that traverse great distances. We present an algorithm for computing and visualizing human centered paths across large datasets, for example the contiguous US. To this end, this work explores graph simplification and various path approximation algorithms in order to create solutions for massive out-of-core data. This work provides an efficient means of generating paths by restricting the search to a subset of the original data. In addition, visualization techniques to compare potential paths, foraging grounds, and alternate destinations are presented.

The application is interactive and allows the user to select start and end locations, as well as one of several path computation algorithms. Satellite imagery is utilized to provide a 93,600 by 212,400 elevation and landcover data grid covering the contiguous US. The available path finding algorithms are Dijkstra's, Fast Dijkstra's (a multi-threaded variant of Dijkstra's introduced in this work), A*, and Single-Query Single Direction PRM (a Probabilistic Road Map algorithm). Most computations are divided into a global and detailed search phase. The global search phase identifies a rough path using a simplified dataset. This

rough path is then used to significantly reduce the total memory and computational time required for the detailed search phase by ensuring that only relevant areas of the terrain are searched. Our application is written in C++ using the OpenGL graphics API and Berkeley Database.

Our results show that path computation over massive out-of-core datasets is possible. We conclude that using our Fast Dijkstra variant provides the best results in terms of accuracy. The contributions of this work include:

- Tools for managing and performing energetic analysis on massive out-of-core datasets. In particular, a restrictive tiling scheme is constructed which significantly reduces the search space without reducing accuracy.
- A multi-threaded bidirectional version of Dijkstra's shortest path algorithm that does not suffer any accuracy loss.
- A comparison between multiple path computation algorithms in terms of runtime, memory usage, and accuracy.
- Visualizations of the terrain from a human traveler's perspective.

2   Previous Work

This work utilizes some of the significant work in the area of path finding algorithms, namely, Dijkstra's [2], A* [3] and Probabilistic Road Maps [4]. Additionally, our application builds on two previous projects involving human centered paths across terrain data, Energetic Analyst [5] and Continuous Energetically Optimal Paths [6]. Brian Wood's Energetic Analyst tool demonstrated the importance of using a human-centered, as opposed to distance-centered, metric for determining the routes of travel for archeology applications. Due to algorithmic constraints, this work could not be applied to large terrain datasets.

The work of Jason Rickwald, Continuous Energetically Optimal Paths (CEP) built on Energetic Analyst while utilizing the Fast Marching Algorithm (FM) [7] for path computations. FM has the benefit of allowing paths to cross a grid face, rather than being constrained to the grid edges. In addition, Rickwald introduced a multi-threaded variant of Fast Marching, which significantly reduced the runtime but introduced a small error in the energetic path computation. Rickwald addressed the memory limitations encountered in Energetic Analyst by providing a mechanism for swapping terrain data between memory and disk. However, algorithmic and data format issues hindered the capability of analyzing very large datasets. For example, a path computation across a dataset covering most of the state of Oregon required almost a full day to compute.

3   Algorithms

A goal of this work is to visualize the terrain and optimal path from one point on the terrain to another. As the dataset for the contiguous United States comprises over 19 billion data points, it was necessary to develop tools to provide a highly simplified, yet acceptably accurate, representation of the dataset for real-time

interaction. This simplification was accomplished by first breaking down the dataset into 434*1,000 non-overlapping rectangular *clusters*. These dimensions were chosen as they maintain the approximate latitude to longitude ratio across the US. As a pre-process, the tool individually loads each *cluster* and performs a simplification on those data points to obtain a single representative data point, using either the average or the median of all the points in the *cluster*.

Path computation occurs when the user selects two arbitrary points on the displayed simplified terrain. The latitude and longitude of these two points is then passed to a variant of Dijkstra's shortest path algorithm that is optimized for the particular graph structure. The resulting path serves as an approximation to the actual path and is visually overlayed onto the simplified terrain. See Figure 1a. To address the problem of the large search space needed with analyzing large datasets, we present a restrictive tiling scheme in which the search space is drastically reduced. First, the full dataset is divided into several hundred *tiles*. Note that these *tiles* are different than the simplification *clusters* as they contain many more datapoints. Next, the *tiles* crossed over by the approximate path generated on the simplified dataset are identified. Finally, the algorithm searches within these *tiles* to construct the detailed path. For robustness, a configurable *buffer* can be set so that if the path falls near the boundary of a *tile*, the neighboring *tiles* can also be searched. This method is successful in limiting the search space without compromising the accuracy of the resulting path. This is a very important feature of our implementation as it prevents a large amount of data swapping that would occur if the path computation was allowed to run un-restricted on the full dataset. See Figure 1a.
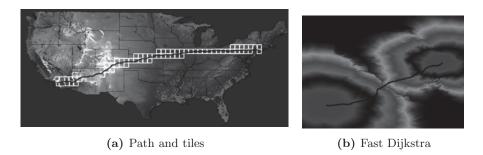

(a) Path and tiles　　　　　　　　**(b)** Fast Dijkstra

Fig. 1. [Algorithm Visualizations] (a) Display of an in progress path computation on the full dataset. The light squares indicate tiles that are within the search space while grey tiles are currently loaded in memory. The simplified path (black) used to determine which tiles to search is also shown. (b) Shows the areas searched by each thread during a run of the Fast Dijkstra algorithm.

In order to compute a human centered optimal path, we must choose graph weights that correspond to the amount of energy it would take a human to traverse that type of terrain. The equations used in this work to determine the caloric cost of travel between two points are the same as for Energetic Analysis

and CEP. These equations were determined and verified under various conditions [8,9]. First, the metabolic rate for traveling between two points is calculated based on the physical parameters of the subject and the slope (grade) of travel. There are two equations, one which computes the metabolic rate when traveling on a positive grade (uphill), while another is used for a negative grade (downhill): See [10] for the exact equations used. For this work, the average velocity when traveling across open ground is $1.34112\frac{m}{s}$, or approximately $4.8\frac{km}{hr}$. When traversing water, it has been experimentally determined that swimming at $0.7\frac{m}{s}$ is roughly equivalent to running at $3.3\frac{m}{s}$ [11]. Unfortunately, these equations can potentially under-predict the caloric cost when traveling downhill at certain velocities. Thus, the computed metabolic rate is compared to the metabolic rate while standing [12], with the larger being used to complete the calculation. The metabolic rate gives the amount of energy expended over time, thus it is necessary to obtain the approximate time required to travel between the two points. Finally, the caloric cost is calculated and converted to kilocalories.

3.1 Path Finding Algorithms

In order to find a path between the source and destination, a number of possible path finding algorithms are provided. The available algorithms are Dijkstra's, Fast Dijkstra's (a multi-threaded variant of Dijkstra's introduced in this work), A^*, and Single-Query Single Direction PRM (a Probabilistic Road Map algorithm). Dijkstra's shortest path algorithm [2] is a well known and extremely pervasive algorithm for determining paths of least cost between two points on a graph. A^* [3] is a slight modification of Dijkstra's algorithm which uses a heuristic to decrease the computation time. A Probabilistic Road Map (PRM) [4] is defined as being a discrete representation of a continuous configuration space generated by randomly sampling the free configurations of a search space and connecting those points in a graph. PRM algorithms are designed for speed at the cost of accuracy, however, due to the probabilistic nature of the algorithms, it is possible to randomly produce an optimal path in a fraction of the time of other algorithms.

PRM: The specific PRM algorithm used in this work is the Single-Query Single Directional PRM (SQPRM) [4]. The idea is to grow a tree type path in random directions until the destination is found. However, since SQPRM's expand in a random fashion, it may require a large amount of time to randomly select and connect the destination node to the graph. Thus, the algorithm terminates when a node is examined that is sufficiently close to the destination node. The detailed pseudo code for the algorithm used for this work is given in [10]. The PRM class of algorithms are designed to quickly construct a traversable path in a large search space, but are not concerned with the actual efficiency of the path. Thus, if a potential edge is not accepted, it can be assumed that that edge will never be added to the graph. However, in the context of energetic paths, it is possible that a previously rejected edge may become viable at a later iteration. The approach taken in this work is to allow a node to be selected and

expanded multiple times, with the cost to reach directly connected (as opposed to all related) nodes being updated when appropriate. Thus, any change to a nodes cost may be slowly propagated as the algorithm progresses. While this does not completely eliminate out-dated information, it does provide a means to reevaluate certain edges and allows the algorithm to include path efficiency as a metric. However, the number of times the node is allowed to be updated can significantly impact the efficiency of the algorithm, which is explained in detail in [10].

Fast Dijkstra's: There has been significant recent work on developing parallelized path computation algorithms [6,13] to utilize the increasing number of cores within standard processors. However, the general problem with these algorithms is that they provide approximations of optimal cost paths, thus sacrificing accuracy for speed. This work presents a bidirectional implementation of Dijkstra's shortest path algorithm which uses two threads to capitalize on modern multi-core processors. This algorithm does not disrupt the optimality properties of Dijkstra's, thus providing optimal paths with a minimal amount of memory overhead. In essence, Dijkstra's algorithm is run separately in two threads with one thread calculating the cost from the start node to the destination node, while the second thread simultaneously calculates the cost from the destination to the start. The two threads meet roughly half-way to their respective goals where one thread is given priority and is responsible for combining the results of the two threads. Care is taken to account for bidirectional graphs, which is important in this work as the caloric cost of traveling uphill differs from traveling downhill as discussed above. Figure 1b illustrates the merging point of the two threads fronts, at the termination of the algorithm. For a more detailed pseudo-code for the fast Dijkstra's algorithm see [10].

4   Results

To demonstrate the tools effectiveness, a number of paths were constructed on massive terrain datasets. All results were obtained on a 2.5 Ghz Intel Core 2 Duo MacBook Pro running OS X 10.5.6 with 4 GB 667 MHz DDR2 SDRAM and a 5400 RPM hard drive.

California Indian Trails: To demonstrate the potential application to the field of Archeology and Anthropology, a trail was plotted between two Native American Indian tribes, one located within a valley between two mountain ranges, and the other located near the coast. California was chosen as a test site as historical records show evidence of healthy trade relations among many of the California Indian tribes. For the exact latitude and longitudes used for the start and stop locations for this example and all others, see [10]. The distance between these two tribes necessitates searching most of southern California. Figure 2a displays the energetic path found by the tools, while Table 1a shows the runtime required for the different algorithms. As shown in this figure, the computed path closely follows the route used and documented by James Davis [14].

As can be seen in Table 1a, using the simplified dataset to get an approximate path and restricting the search space (with a small buffer) on the detailed dataset can still produce a perfectly accurate path. Notice that the required time is drastically reduced for both the $Dijkstra's$ and $Fast\ Dijkstra$ variant when using restrictive tiling, but with no error in the path. In addition, A^* provides a path with very little error while requiring even less time than the $Fast\ Dijkstra$ algorithm. However, PRM took a substantially longer amount of time to complete and provided a highly inaccurate path. This does not conclusively determine the inappropriateness of PRM as the probabilistic nature of it means results may vary between runs.

Table 1. Path computations: Note that '$Dijk \rightarrow Dijk_{Fast}$' indicates that Dijkstra's was used on the full dataset to determine which tiles to search using the Fast Dijkstra algorithm. For each result, the error percentage is based on the difference in cost between the indicated method and the cost obtained from running Dijkstra's unrestricted on the full dataset (marked with a *). *Nodes* indicates the number of data points that were analyzed. All costs are in kilocalories.

Method	Nodes	Runtime	Memory	Cost	Error
*$Dijk$	265,024,564	1h 21m 3s	2.36 GB	42,455.5	0.00%
$Dijk \rightarrow Dijk$	92,548,510	23m 37s	1.35 GB	42,455.5	0.00%
$Dijk \rightarrow Dijk_{Fast}$	100,371,403	14m 26s	1.51 GB	42,455.2	0.00%
$Dijk \rightarrow A^*$	39,576,328	11m 31s	1.35 GB	42,880.5	1.00%
$Dijk \rightarrow PRM$	77,423,841	42m 3s	1.19 GB	54,417.4	28.17%
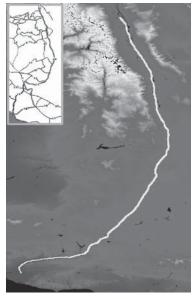
(a) California Indian Trail

Method	Nodes	Runtime	Memory	Cost	Error
*$Dijk$	505,896,251	2h 41m 50s	2.34 GB	74,653.2	0.00%
$Dijk \rightarrow Dijk$	184,985,532	46m 5s	1.80 GB	74,653.2	0.00%
$Dijk \rightarrow Dijk_{Fast}$	181,351,914	26m 9s	1.95 GB	74,655.7	0.00%
$Dijk \rightarrow A^*$	164,865,746	47m 54s	1.95 GB	75,776.5	1.50%
$Dijk \rightarrow PRM$	164,464,736	2h 22m 29s	1.94 GB	97,402	30.47%

(b) Between Old Fort Boise, ID and Oregon City, OR

Method	Nodes	Runtime	Memory	Cost	Error
*$Dijk$	735,795,927	3h 18m 39s	2.36 GB	107,723	0.00%
$Dijk \rightarrow Dijk$	252,449,787	1h 14m 59s	1.94 GB	107,723	0.00%
$Dijk \rightarrow Dijk_{Fast}$	285,154,704	37m 15s	1.95 GB	107,726	0.00%
$Dijk \rightarrow A^*$	230,861,134	1h 12m 18s	1.91 GB	112,149	4.11%
$Dijk \rightarrow PRM$	185,307,75	2h 24m 31s	1.94 GB	123,435	14.59%

(c) Between Moundville, AL and Hopewell, OH

(a) California Indian Trai   (b) Oregon Trail - Detailed View

Fig. 2. (a) An energetic path, possibly corresponding to the trails mapped by James Davis [14] shown in the upper right. (b) An energetic path overlayed with an estimate of the historic Oregon Trail [15].
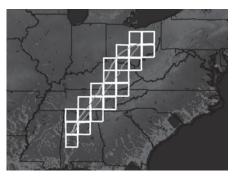
Oregon Trail: To demonstrate a comparison against previous work while demonstrating a historical application, a path was computed following the Oregon Trail, which is a well known trail taken by settlers journeying to the western United States in the mid 1800's. This example begins at Old Fort Boise in Idaho and ends at Oregon City in Oregon. Figure 2b shows the paths generated on the full dataset, while Table 1b presents the results. As clearly demonstrated in Figure 2b, the computed path closely follows the historical trail.
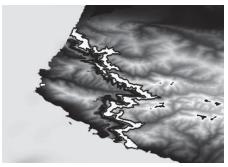
Constructing a path across Oregon using previous work, CEP, required most of the day. By contrast, the restrictive tiling scheme presented in this work significantly reduces the amount of time to typically less than an hour while still producing accurate paths. The results are comparable to those of the California Indian Trail in that both the *Dijkstra* and *Fast Dijkstra* methods produced accurate paths.

Moundbuilders: To demonstrate a path computation over a significant distance, an energetic path is computed that may have been used between two prominent Moundbuilder villages; Moundville, Alabama and Hopewell, Ohio. Ancient burial and ceremonial mounds constructed by the Moundbuilders have been found all along the central and eastern United States. Excavation from these mounds have revealed flint from the Rocky Mountains, shells from the Gulf of Mexico, and other artifacts of distant origin [16]. This indicates that the

inhabitants engaged in extremely long-distance trade, although the actual trade routes remain somewhat of a mystery. Figure 3a visualizes the least cost caloric path between the two sites and the results are presented in Table 1c.

This longer path required increased runtimes and a larger number of nodes to be searched for each method. Notice that, compared to the results for the previous paths, PRM is able to construct a relatively accurate path in a small fraction of the time required by $Dijkstra$ when run unrestricted on the full dataset. However, $Fast\ Dijkstra$ is still able to produce a far more accurate path in similar time while consuming considerably less memory.

(a) Moundbuilders - Hopewell to Moundville

(b) CA-SLO-9

Fig. 3. (a) A possible trade route utilized by the Moundbuilders. While the paths from the simplified (dark grey) and full (light grey) datasets differ significantly, the simplified path is still sufficient to determine which full dataset tiles to search. (b) California archaeological sites with the caloric radius visualization. Likely boundary of foraging region is banded black and white regions. The dark body of water seen at the top of the image is Morro Bay in California.

California Archaeological Sites: In the archaeological community, a traditional means of determining a tribes hunting and foraging grounds is to draw a circle on a map, focused on the tribes camp, with a radius of several kilometers. However, this method does not account for the terrain type. For example, a hunting party can travel much further on flat terrain (thus extending the radius) as opposed to very rocky or sloped terrain. Thus, a better metric for determining a tribes hunting ground may be a caloric measure. Figure 3b shows archaeological sites in central California, designated CA-SLO-9 respectively. The caloric radius visualization shows the area for which the tribes located at these sites may have searched for food. Notice that when the elevation remains level, such as along the coast or through valleys, a much greater distance can be reached, while the converse it true for hilly regions.

Visualizations: In addition to the specific results examples, the system allows for visualizations for further analysis of the paths and the terrain from the

perspective of a walking human. Figure 4 demonstrates potential paths that end near the original destination and neighboring areas on a path that could be traversed without adding significant cost to the original path. These types of visualization are intended to help visualize other potential regions of interest when searching for a historical path.

(a) Nearby Paths: Black path indicates an alternate path

(b) Path Regions: Reachable areas (shown as light regions near the path), are drastically decreased through valleys

Fig. 4. Nearby path visualizations

Notes on the Results: Note that the results occasionally show a slight difference in cost between Dijkstra's shortest path algorithm and the Fast Dijkstra variant. This is caused by floating point rounding differences during the caloric cost calculations, not an actual difference in paths produced. Also note that the results provided for the PRM algorithms are for a specific run. Due to the probabilistic nature of the algorithms, each run will likely produce a different result. In addition, A* generally utilizes an admissible heuristic function to estimate the cost from the current node to the destination. Admissible indicates that the heuristic must not overestimate the cost to the destination. For this work, the heuristic uses the Euclidean distance from the current node to the destination, combined with the metabolic rate associated with the corresponding grade, to estimate the total caloric cost. Unfortunately, this heuristic is not admissible in certain rare cases, which introduces a small error into some path computations.

5 Conclusions and Future Work

We present a set of tools that can be used to analyze massive out-of-core terrain datasets. We have demonstrated the efficiency and possible historical applications of our tools by performing several experiments to construct energetic paths across large distances using a variety of algorithms. Using our multi-threaded Dijkstra variant combined with restrictive tiling, we are able to construct an accurate energetic path across the United States in under three hours – a large

improvement over previous work in which a path across the state of Oregon required most of a day.

Avenues for future work include exploring alternate datasets from around the world. Additionally, more advanced simplification methods may provide more accurate paths, allowing for the use of smaller tiles to further reduce the search space. Also, PRM optimizations could be explored, such as utilizing the fast runtime by running PRM multiple times and presenting the best result or a bidirectional approach for further speed increases.

References

1. Jones, T.: Personal Correspondence (2009)
2. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
3. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics 4, 100–107 (1968)
4. Hsu, D., Claude Latombe, J., Motwani, R.: Path planning in expansive configuration spaces. International Journal of Computational Geometry and Applications, 2719–2726 (1997)
5. Wood, B.M., Wood, Z.J.: Energetically optimal travel across terrain: visualizations and a new metric of geographic distance with anthropological applications. In: SPIE, vol. 6060, p. 60600 f. (2006)
6. Rickwald, J.: Continuous energetically optimal paths across large digital elevation data sets. Master's thesis, California Polytechnic State University, San Luis Obispo (2007)
7. Kimmel, R., Sethian, J.A.: Computing geodesic paths on manifolds. Proc. Natl. Acad. Sci. USA, 8431–8435 (1998)
8. Duggan, A., Haisman, M.F.: Prediction of the metabolic cost of walking with and without loads. Ergonomics 35, 417–426 (1992)
9. Pandolf, K.B., Givoni, B., Goldman, R.F.: Predicting energy expenditure with loads while standing or walking very slowly. Journal of Applied Physiology 43(4), 577–581 (1977)
10. Tsui, A.N.: Energetic path finding across massive terrain data. Technical Report CPSLO-CSC-09-02, Department of Computer Science, California Polytechnic State University, San Luis Obispo, California (2009)
11. Prampero, P.E., Pendergast, D.R., Wilson, D.W., Rennie, D.W.: Energetics of swimming in man. Journal of Applied Physiology 37, 1–5 (1974)
12. Harris, J.A., Benedict, F.G.: A biometric study of basal metabolism in man. Cornell University, Mann Library, Ithaca, New York (1919)
13. Weber, O., Devir, Y.S., Bronstein, A.M., Bronstein, M.M., Kimmel, R.: Parallel algorithms for approximation of distance maps on parametric surfaces. ACM Trans. Graph. 27, 1–16 (2008)
14. Davis, J.T.: Trade Routes and Economic Exchange Among the Indians of California. University of California Archaeological Survey, CA (1961)
15. City, H.O.: End of the oregon trail interpretive center (2009), http://www.historicoregoncity.org/HOC/index.php?view=article&id=57
16. Fagan, B.M.: From Black Land to Fifth Sun: The Science of Sacred Sites. Addison-Wesley, Reading (1998)