Spatio-temporal Matching Algorithms for Road Networks[*]

Daniel Ayala, Ouri Wolfson, Bo Xu,
Bhaskar DasGupta
University of Illinois at Chicago
Department of Computer Science
dayala@uic.edu, {wolfson, boxu,
dasgupta}@cs.uic.edu

Jie Lin
University of Illinois at Chicago
Department of Civil and Materials Engineering
janelin@uic.edu

ABSTRACT

In this paper we present a model of spatially located mobile agents and static resources, in which the agents are looking to obtain one of the resources while minimizing their costs to obtain the resource. The proliferation of mobile devices, location-based services and embedded wireless sensors has given rise to applications that could help the mobile agents have updated information of the location of the resources they are looking for. Nevertheless, while engaged in driving, travelers are better suited being guided to an ideal resource, rather than looking at a map and deciding which available resource to visit. Then the question of how an application should choose this ideal resource, to guide the agent towards it, becomes relevant. In this work we develop algorithms that are designed to guide users to these resources. They use a gravitational approach to guide a mobile agent through a road network in order to find this ideal resource. The performance of the algorithms is evaluated through simulations.

Categories and Subject Descriptors

J.m [**Computer Applications**]: Miscellaneous

1. INTRODUCTION

In this paper we consider a matching problem between *mobile agents* and *static resources* that are located on a grid. The movement of the agents is also constrained to the grid. Each agent is looking to obtain one of the resources and they make routing choices in order to obtain it.

This type of model applies to situations that occur in the urban transportation system. For example, consider the situation where there are a set of taxicabs (mobile agents) looking for clients (static resources) to pick up. Also, in parking, vehicles (mobile agents) are looking to obtain an available parking slot (static resources). The resources are all of the same nature, *e.g.* all are clients (in the taxi application) or all are parking slots (in the parking application).

These types of applications rely on having available the location information of the static resources. The proliferation of mobile devices, location-based services, and embedded wireless sensors make it possible for mobile agents to have information of the locations of the potential resources that they want to gain [4, 8]. For example, taxicabs could have a system where potential clients request a cab and the locations of said clients are displayed on a map for the cab driver to see. For the parking problem, new applications are arising that help travelers find parking in urban settings. Wireless sensors that are embedded on parking slots are used to detect the availability of slots across some area. Then the locations of currently available parking slots are disseminated to the mobile devices of users that are looking for parking in the area. A prime example of this application is SFPark [3]. It uses sensors embedded in the streets of San Francisco. When a user is looking for parking in some area of the city, the application shows a map with the marked locations of the open parking slots in the area.

While these types of systems, described in the above paragraph, could prove useful to the mobile agents in their quest to obtain their desired resource, they do raise safety concerns for the drivers. Drivers have to shift their focus from the road to the map being displayed by the mobile device they are using. Then they have to study the map and choose what is their most preferred resource to visit and try to obtain it. It would be safer if the application simply guided each agent to the location of an *ideal* resource for him/her. Then the question arises, which algorithm should a mobile application use to choose such an ideal resource?

Regardless of the safety concerns stated in the previous paragraph, the question still remains relevant. What is the optimal way of moving towards spatially located resources, to obtain one of the resources, when there is competition for the resources (from other agents that also want it)? What is the best way that a mobile agent can be guided towards an ideal spatial resource?

We answer this question in an incomplete information context, a competitive model in which the agents are not aware of the locations of the other agents. For this model, the gravity-based algorithms presented in [1] and [2] are suitable. Nevertheless, [1, 2] used a 2-dimensional free (i.e. Euclidean) space model. Here we extend those algorithms to a more realistic scenario, i.e. road networks. In other words, we consider a gravity-based approach in a road-network, where the movement of agents is constrained to road seg-

[*]This work was supported in part by the NSF IGERT program under grant DGE-0549489.

ACM SIGSPATIAL GIS '12 November 6-9, 2012. Redondo Beach, CA USA
Copyright 2012 ACM ISBN 978-1-4503-1691-0/12/11 ...$15.00.

ments. We propose three possible approaches to adapt the gravity-algorithm to road-networks: the Deterministic Angular Gravity-based spatial Resource selection Algorithm (DA-GRA), the Randomized Magnitude GRA (RM-GRA) and the Deterministic Magnitude GRA (DM-GRA). The DM-GRA shows improvements of up to 47% compared to a tested greedy algorithm.

2. GENERAL SETUP AND NOTATION

The general setup of our spatio-temporal matching problem in a road network is as follows:

- There are two types of *objects* as follows.
  - A set of n mobile agents $A = \{a_1, a_2, \ldots, a_n\}$.
  - A set of m available static resources $R = \{r_1, r_2, \ldots, r_m\}$.
- A road network is modeled as a directed graph $G = (V, E)$ with the nodes of the graph being the intersections in the road network and the edges being the road segments between the intersections.
- The road network is embedded on Euclidean space and the locations of the agents and the resources are restricted to being on points in the road segments.
- Each agent is assumed to be moving independently of all other agents at a fixed velocity. Without loss of generality, we assume that the *moving speeds* of all agents are the same[1].
- $\mathsf{time} : A \times R \to \mathbb{R}$ is a distance function given in terms of time. It denotes the current travel time between an agent and a resource. It is the travel time along the shortest path between the agent and the resource.
- $\mathsf{cost} : A \times R \to \mathbb{R}$ is a cost function. It denotes the *cost* of a resource $r_j \in R$ to an agent $a_i \in A$. This cost is a general cost. It will be dependent on the application being considered. For example, in the parking application, the agents will care about the time it takes to reach their destinations. So then in that case the cost will be a function of the travel time (time) to the parking slot and possibly the walking time to their destination from the given parking slot (resource). Additional cost components could be the dollar price of the slot, the safety of the area, etc.
- Each agent can only make a routing decision upon arrival to an intersection, *i.e.* a user cannot change the course of its current direction until it reaches an intersection. This is due to the constraints imposed by movement on the grid. Once a driver has entered a road segment, he can only get out of the segment by reaching the end of it (the intersection). Our algorithms for the road network will only make routing decisions at intersections.
- An agent looks to make an individual decision on which resource to move towards, that will help them minimize their *cost* when traveling to an obtained resource.
- Agents are assumed to have updated information of resource availability at all times. This means that if while heading to its resource of choice, the resource is taken by another agent, then the agent will be informed that the resource is no longer available. Then upon reaching its upcoming intersection, the vehicle can make a new resource choice according to the currently available resources. Another situation where a vehicle may want to update its resource choice is if new resources become available that are preferrable.

[1]Otherwise, we simply need to rescale the distances for each agent in our algorithmic strategies.

We categorize this setup as a Spatio-temporal matching problem on a road network. The *spatial* component of the matching comes from the locations that the agents and resources have on the map and from the distances between them. The *temporal* component of the matching comes from the time it takes for an agent to reach the resource it has been matched with (or obtains). The dynamic nature of the problem considered here adds another temporal aspect since the objects (agents and resources) have start times.

3. GRAVITY-BASED SPATIAL RESOURCE SELECTION ON A ROAD NETWORK

The Gravity Parking Algorithm (GPA) was introduced in [1] to guide agents towards ideal areas of the map when they do not have information about the other agents that are competing with them for the resources. The GPA was designed for parking applications but it can be applied to the general framework we are presenting in this work. In the GPA for 2D free-space, resources are said to have a gravitational pull on the agents. At any point in time, each resource has a gravitational force on the agent that will depend on the distance from the agent (magnitude) and location of the resource (direction). The intuition is that the agent will be pulled towards areas with a higher density of resources, thus increasing the probability of finding an available resource close to the area that the agent is driving through. So then for each resource, a force vector is generated pulling the agent. Then, all of these vectors are added and the agent moves in the direction of the resultant vector (total gravitational force) for a specified time step. Then the process is repeated at the beginning of each time interval.

A simplified formula for gravitational force was used to generate these gravity vectors, which did not include the masses of the objects or a gravitational constant like in the classical gravitational force equation. This formula was:

$$F(a, r) = 1/\mathsf{cost}(a, r)^2 \tag{1}$$

With formula 1, one will compute gravitational pull by considering the general cost as the distance between the agent and the resource.

This method was shown to work well in 2D free-space. On the embedded road network, we will still use a gravitational approach. It will also be based on the gravity formula defined by equation (1).

An agent can only make a routing choice upon arrival to an intersection, whereas before (in free-space) an agent could change direction at any point in time. Therefore, the Gravity-based spatial Resource selection Algorithm (GRA) will only be used at each intersection by each agent.

Instead of adding up all the gravity vectors for all resources (as in Euclidean space), the agent will aggregate the gravity information for all resources into special direction vectors (one for each possible direction out of the intersection). Suppose that the intersection where agent a is located has k outgoing edges $e_1, e_2, ..., e_k \in E$. Then there will be k direction vectors $g_1, g_2, ..., g_k$ where each vector will have a direction according to its respective embedded edge. The magnitudes of these vectors will start at 0.

Then for each available resource $r \in R$, the shortest path is computed from a to r and the gravity force g is computed using equation (1). Let e_i be the first edge to be taken according to the computed shortest path. Then g_i is updated to be $g_i = g_i + g$.

After repeating this procedure for each resource, the agent will use the computed direction vectors $g_1, g_2, ..., g_k$ to make its route choice.

From this point, we will introduce three variants of the GRA that will be evaluated as candidate algorithms. The three variants will only differ in how the eventual edge to be taken is computed based on the direction vectors.

3.1 Deterministic Angular GRA (DA-GRA)

In the Deterministic Angular GRA (DA-GRA) the direction vectors $g_1, g_2, ...g_k$ will be added to produce a resultant vector v. This resultant vector will be located between two of the directions to choose from, say $e_i \in E$ and $e_j \in E$. Let θ_i be the angle distance between v and e_i and θ_j be the angle distance between v and e_j. Then, if $\theta_i < \theta_j$, a will choose e_i as the next edge to travel, otherwise it will choose e_j as the next edge to travel through.

3.2 Randomized Magnitude GRA (RM-GRA)

In the Randomized Magnitude GRA (RM-GRA) the direction vectors $g_1, g_2, ..., g_k$ will be used as part of a probabilistic scheme. Let $T = |g_1| + |g_2| + ... + |g_k|$, *i.e.* the addition of the magnitudes of the k direction vectors. Then let $p_i = |g_i|/T$ for $1 \le i \le k$. Then each edge $e_i \in E$ which is an outgoing edge of a's current intersection will be chosen with probability p_i.

3.3 Deterministic Magnitude GRA (DM-GRA)

In the Deterministic Magnitude GRA (DM-GRA) the direction vectors $g_1, g_2, ..., g_k$ will be used to choose the next direction to move towards. The direction with the vector with the largest magnitude will be chosen.

The efficiency of these three GRA variants will be evaluated through simulation. The simulation setup and the results are presented in the next section.

4. SIMULATION AND RESULTS

In this section we will evaluate DA-GRA, RM-GRA, and DM-GRA against a greedy resource selection algorithm. The greedy algorithm simply moves each agent towards its current closest available resource.

4.1 Simulation Environment

The simulation evaluates the GRA variants with varying numbers of vehicles and resources. The simulation is run on a 1 mile by 1 mile map where roads are generated that either run from east to west or north to south. This is a road network with 10 roads that have North/South directions and 10 roads with East/West directions. They form a perfect grid with the intersections of these roads as nodes.

In reality, the resources are not uniformly distributed. Thus, we generate skewness as follows. The map is partitioned into 16 equal-sized square regions. A random permutation of the regions is generated (uniform distribution) and is used as the ranking of the popularity of each region for available resources. To choose the location of each of the m available resources, first a random number is generated to determine in which region to place the resource. The Zipf distribution with its skew parameter and the regional popularity previously generated are used to generate this random number. Then a random position in the grid (uniform) is chosen from the region denoted by the Zipf number. The n agents' initial positions are generated using the uniform distribution on the grid.

After generating the agents and resources, the algorithms are tested. The GRA algorithms were tested against the greedy resource selection algorithm, which simply moves each agent towards its current closest resource. For the GRA algorithms, the agents will move as described in the procedures described in section 3.

When an agent reaches an available resource, the time it took for it to find that resource is saved. Then a new resource is generated on a randomly chosen region (Zipf). Also a new agent is generated at a random location on the grid (uniform). This way the number of agents and resources (thus the level of competition for resources) is kept constant. The simulation run stops when a given time horizon of 3,600 seconds is surpassed.

The parameters of the simulation and the values that were tested for each parameter are detailed in table 1. For each configuration of the parameters, 100 different simulation runs were generated and tested.

Parameter	**Symbol**	**Range**
Agents	n	{40,80}
Resources	m	{20,30,40} when $n = 40$ {40,60,80} when $n = 80$
Regional Skew	-	{0, 1, 2, 3}
Velocity	v	{10,15,20,25,30} in mph

Table 1: Parameters tested on Simulation

4.2 Simulation Results

We first evaluate the GRA algorithms against the greedy approach by using the cost as time traveled towards the slot ($\mathsf{cost} = \mathsf{time}$). With this type of analysis we can assess the potential reductions in driving time (assuming the agents are traveling in vehicles) by using the GRA algorithms. Reductions in driving time lead to benefits for the environment in reductions of gasoline waste and CO_2 emissions.

Figure 1 shows the results of how much percent improvement was obtained by the GRA algorithms over the greedy resource selection algorithm for 80 agents and resources. We see that the GRA algorithms perform the worst when the regional skew is 0. In this case the resources are distributed uniformly across the grid. The DA-GRA and RM-GRA variants have a negative improvement in this case of regional skew being 0. However, the DM-GRA gets a positive improvement over the greedy algorithm even in this case.

The GRA algorithms perform better when the skew is higher than 0. This is to be expected because when resources are co-located regionally (because of higher regional skew), the GRA algorithms will seek to push agents towards areas where the density of available resources is higher. Although the performance of the GRA algorithms started decreasing when the regional skew increased past 1, they did have improved performance over the case where the resources are uniformly distributed. We also tested the GRA algorithms with varying number of available resources (m). The results

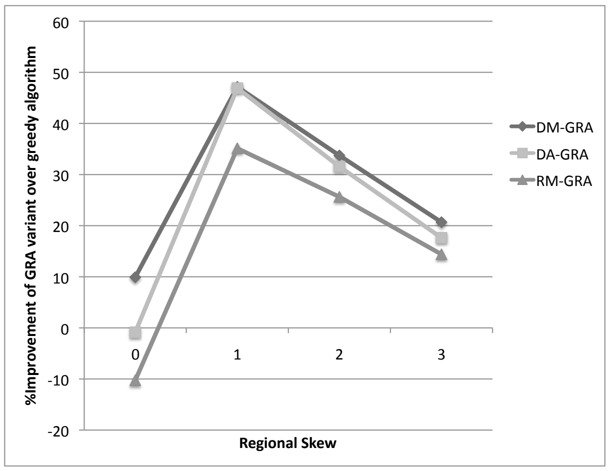

Figure 1: % Improvement of GRA algorithms over Greedy (80 agents, 80 resources)

were very similar to those reported in figure 1.

Also, the results with a higher number of agents, $n = 80$, were better than with $n = 40$. The highest improvements for the GRA algorithms were obtained in these test cases ($n = 80$). The highest improvement was over 47% compared with the greedy resource selection algorithm in figure 1 ($n = 80, m = 80$, skew $= 1$).

From the obtained results we can also attest that DM-GRA is the GRA algorithm of choice. It obtained a positive improvement in all test cases. Also, in all cases it obtained the optimal improvement, amongst the GRA variants, or very close to the optimal (it never deviated more than 1% from the optimal). Like the DM-GRA, the DA-GRA variant performed well. However, in cases where the regional skew was 0, the DA-GRA did not obtain a positive improvement.

The RM-GRA, a randomized algorithm, did not perform as well in the simulations as its deterministic counterparts. In cases where the resources were uniformly distributed (skew = 0) it was outperformed significantly, even by the greedy parking algorithm. The unexpected results for the RM-GRA are perhaps due to the inherent cycling that can occur with random walks of directed graphs.

The best percent improvement that was obtained by the DM-GRA over the greedy algorithm was of around 47% (n = 80, m = 80, skew=1). Consider the parking problem in which the agents are vehicles looking for open parking slots (resources). In [5], studies conducted in 11 major cities revealed that the average time to search for curbside parking was 8.1 minutes and cruising for these parking slots accounted for 30% of the traffic congestion in those cities. This means that each parking slot would generate 4,927 vehicle miles traveled (VMT) per year [6]. That number would of course be multiplied by the number of parking slots in the city. For example, in a big urban city like Chicago with over 35,000 curbside parking slots [7], the total number of VMT becomes 172 million VMT per year due to cruising while searching for parking. Furthermore, this would account for waste of 8.37 million gallons of gasoline and over 129,000 tons of CO_2 emmisions.

Then the 47% improvement shown by the DM-GRA for the time taken to find a parking slot (proportional to the tested distance traveled because of constant velocity), would reduce vehicle miles traveled for a city like Chicago by over 80 million VMT. This gives a reduction of over 3.9 million gallons of gasoline and of over 60,000 tons of CO_2 emissions per year in a big city like Chicago.

We also ran simulations by considering a generalized cost for the parking problem (driving+walking time). The results showed positive improvements in all test cases for the DM-GRA. The results are omitted for space considerations.

5. CONCLUSION

In this paper we presented a model of spatially located mobile agents and static resources, in which the agents are looking to obtain one of the resources while minimizing their costs to obtain the resource. The main goal of this work was to present a Gravity-based Resource selection Algorithm (GRA) that could be implemented with movement constrained to a road network. This algorithm will be used by a mobile application to guide an agent to an ideal area of the map where he is most likely to find a spatially located resource. We presented three potential gravity-based algorithms for evaluation: the Deterministic Angular GRA (DA-GRA), the Randomized Magnitude GRA (RM-GRA) and the Deterministic Magnitude GRA (DM-GRA). The merits of the proposed algorithms were tested through simulations.

The DM-GRA was the best performing algorithm, and always improved over the greedy resource selection algorithm which simply moves agents to their closest available resources. It showed improvements of more than 47% for some test cases. This yields a reduction of over 3.9 million gallons of gasoline and over 60,000 tons of CO_2 emissions per year in a big city like Chicago when the algorithm is applied to parking situations.

6. REFERENCES

[1] D. Ayala, O. Wolfson, B. Xu, B. Dasgupta, and J. Lin. Parking slot assignment games. In *Proc. of ACM SIGSPATIAL GIS 2011*, Chicago, IL, November 2011.

[2] D. Ayala, O. Wolfson, B. Xu, B. DasGupta, and J. Lin. Parking in competitive settings: A gravitational approach. In *Proc. of MDM 2012*, Bengaluru, India, July 23-26 2012.

[3] http://sfpark.org/.

[4] R. Kühne. Potential of remote sensing for traffic applications. In *Proceedings of IEEE Intelligent Transportation Systems*, Oct. 12-15, 2003.

[5] D. Shoup. *The High Cost of Free Parking.* American Planning Association, Chicago, IL, 2005.

[6] D. Shoup. Cruising for parking. *Transport Policy*, 13:479–486, 2006.

[7] Transportation Alternatives (www.transalt.org), New York, NY. *Pricing the Curb: How San Francisco, Chicago and Washington D.C. are reducing traffic with innovative curbside parking policy*, July 2008.

[8] O. Wolfson and B. Xu. Opportunistic dissemination of spatio-temporal reseource information in mobile peer-to-peer networks. In *Proc. of 1st International Workshop on P2P Data Management, Security and Trust (PDMST'04), DEXA Workshops 2004*, pages 954–958, Zaragoza, Spain, Sept. 2004.