

# Real-Time Kinetic Algorithms

Patchrawat “Patch” Uthaisombut

Department of Computer Science, University of Pittsburgh.

email: utp@cs.pitt.edu, url: www.cs.pitt.edu/~utp

The *real-time kinetic (RK)* model for studying kinetic problems is introduced. The problem of maintaining an approximate ordering of moving points on the line over time is studied. We give algorithms that have small geometric errors for this problem. This work should be viewed as an initial investigation into a research area where there appear to be many interesting problems at the intersection between the discrete and the continuous.

In a *kinetic* problem, there is a system that contains real-valued parameters and a function on these parameters. The combinatorial description of the value of the function at any time is called the *configuration* of the system. The *value* of the configuration is the value of the function that the configuration describes. The parameters of the system change continuously over time. As the parameters change, the configuration changes. The goal of the problem is to maintain the configuration (or a close approximation) over time. The challenge is to find the most efficient way to do this by exploiting fact that the parameters change in a continuous manner but the configuration only changes at discrete times. An example of a kinetic problem is a system of points on the plane that move continuously over time, and the objective is to maintain a minimum spanning tree (MST) over time. The *configuration* here is the set of edges of an MST. The *value* of the configuration is the total weight of the edges in the MST.

The *real-time kinetic (RK)* model for studying kinetic problems is introduced. This model has the following features. We assume that the speed of any object at any time is no greater than a certain maximum speed  $S_{\max}$ . An object can move on an arbitrary trajectory. Furthermore, the algorithm learns the position of the objects in an online fashion, that is the algorithm does not know and cannot predict with certainty the position of an object in the future. The algorithm must maintain the configuration in real time, that is, it performs computation at the same time that the parameters of the system are changing. Generally the optimal configuration cannot be maintained at all time under reasonable assumptions. Thus, we settle for an approximate configuration. In such cases, we are interested in the quality of the *value* of the configuration, *i.e.* how close it is to the value of the optimal configuration.

Kinetic problems have been studied under different assumptions. Atallah [1] studied the case where the complete trajectory of each object is known in advance. *Kinetic data structures (KDS)* was introduced by Basch, *et.al.* [2]. In the KDS model, it is still assumed that trajectories of objects are known. However, the trajectories can be updated over time in an online fashion, that is, the algorithm does not know when trajectories will change. In contrast, in the RK model, the algorithm learns the trajectories of the objects in a completely online fashion. In the KDS model, two of the goals are to minimize the *amount of computation per event* to maintain the certificates (*responsiveness*) and the ratio of the number of internal and external events (*efficiency*). In contrast, the goal in the RK model is to minimize the *maximum absolute error* over time.

Algorithms in the KDS model are not meant to run in real time. Therefore, the issue on degeneracy is left untouched. If many events occur in a short period of time, there is no guarantee on the timeliness or the quality of the solution. The KDS model is appropriate for non-real-time applications where objects move under some known law, but some events may cause the trajectory of objects to change. An example is the simulation of billiard balls colliding each other. In the RK model, algorithms run in real time and deal with degeneracy directly. The RK model is appropriate for real-time monitoring of real physical systems that evolve in an unknown or unpredictable way. Examples are tracking an autonomous moving object in a sensor network and maintaining an MST

of nodes in a mobile ad hoc network.

To illustrate the model, we show preliminary results for the problem of maintaining an approximate ordering of moving points on the line over time. Our algorithm maintains a data structure that supports *search by rank* and *nearest neighbor search*. Given a rank  $k$ , the search-by-rank operation initiated at time  $t$  should return the identity  $i$  of the object of rank  $k$  at time  $t$ . Suppose  $v_i(t)$  is the position of object  $i$  at time  $t$ . The *error* of a search by rank is defined to be  $|v_i(t) - v_j(t)|$  where  $j$  is the object returned by the search operation.

We give an algorithm that has a maximum error of  $O(S_{\max}n)$ . Note that the naive method of running a static sorting algorithm repeatedly over time has a maximum error of  $\Omega(S_{\max}n \log n)$ . The idea of our algorithm is to group objects that are close together into clusters and to maintain the ordering of clusters instead of the objects themselves. All members of a cluster is represented by a representative of the cluster. By choosing the right size for the clusters, it is possible to remove degeneracies while maintaining a small error bound.

If the objects do not congregate together too much, then with some additional assumptions, it is possible to lower the absolute error even more. Suppose  $P$  is a set of stationary objects on the line. We say that objects  $i$  and  $j$  are  $\varepsilon$ -close to one another if  $|v_i - v_j| \leq \varepsilon$ . A set  $D$  is an  $\varepsilon$ -dominating set for  $P$  if for any point  $x$  in  $P$ , either  $x$  is in  $D$  or  $x$  is  $\varepsilon$ -close to a point in  $D$ . A *minimum  $\varepsilon$ -dominating set* for  $P$  is an  $\varepsilon$ -dominating set with the smallest cardinality. The cardinality of such sets is denoted by  $\lambda_\varepsilon(P)$ . The *degree of  $\varepsilon$ -congregation* of  $P$ , denoted  $\gamma_\varepsilon(P)$ , is defined to be  $n - \lambda_\varepsilon(P)$ . Note that the value of  $\gamma_\varepsilon(P)$  is between 0 and  $|P|$  inclusive. We show that if  $\gamma_\varepsilon(P) \leq k$  where  $\varepsilon = S_{\max}k$ , then there is an algorithm that has a maximum error of  $O(S_{\max}k)$ .

This work should be seen as an initial investigation into a research area where there appear to be many interesting problems at the intersection between the discrete and the continuous. Another interesting feature of the model is that an algorithm should not spend too much time computing, because the objects are moving at the same time. The more time it uses, the more its result will degrade. This is in contrast to most computational models where the more time an algorithm spends doing computation, the better the solution.

The real-time kinetic (RK) model can be investigated in many directions.

- Many static problems can be naturally extended to kinetic ones and studied under the RK model, and this is wide open.
- Suppose we want to keep track of only the highest object over time, it is unclear if the maximum error can be lower than  $O(S_{\max}n)$ . Consider the case where all  $n$  objects are initially locate at the same position, and then suddenly an object or a set of objects breaks away.
- Our algorithms need to know the maximum speed of the objects *a priori*. Is there an algorithm that adaptively varies its error rate according to the current speed of the objects?
- In many applications, partial knowledge of objects trajectories is known. How to incorporate this partial knowledge? How to combine the KDS and the RK models?

## References

- [1] M.J. Atallah. Some dynamic computational geometry problems. *Computers and Mathematics with Applications*, 11:1171–1181, 1985.
- [2] Julien Basch, Leonidas J. Guibas, and John Hershberger. Data structures for mobile data. *Journal of Algorithms*, 31(1):1–28, 1999.