A FAST RESPONSE SYSTEM FOR AEROSOL DISPERSION FORECASTING

Andrei Smirnov and Steven Rowan
West Virginia University
Morgantown, WV

ABSTRACT
A technique for express analysis of aerosol dispersion in urban environments is presented. In the proposed method a database of different aerosol dispersion scenarios is generated in extensive computer simulations on distributed computing platforms, such as clusters or grid environments. The data is segmented into smaller chunks, and compressed for optimal network access from remote locations. The results of the simulations can then be retrieved in a timely manner for express risk analysis. A prototype solver was developed and tested. Fast response of the system is achieved by replacing complex 3D simulations with information retrieval from a remote database. The reduction of system response from a few days to several seconds can be achieved compared to real-time 3D simulations.

Keywords: Physically based modeling, Grid computing, Web Interfaces, Risk analysis, Information Retrieval, Aerosols

1 Introduction

In the years following the terrorist attacks upon the United States on September 11, 2001, there has been a greater emphasis placed upon the preparation for, and prevention of, another attack. One of the scenarios considered by organizations such as the Department of Homeland Security, Center for Disease Control and the Environmental Protection Agency (to name a few) is the detonation of a Radiological Dispersion Device (a so-called dirty bomb) or the release of an aerosol-based biochemical or biological agent such as Anthrax, Sarin or Ricin within an urban environment.

According to information available via its website, the U.S. Department of Homeland Security has sponsored planning scenarios in order to help prepare municipalities for the possibility of such an event. In addition, researchers at the Army High Performance Computing Research Center in Minneapolis, MN have been working on using computational fluid dynamics in order to model the spread of an airborne contaminant in an urban environment using Cray X1E supercomputers [1]. While these simulations can provide very accurate results, the time required to run a simulation of a specific incident might very well take from several hours to several days to complete, depending upon the size of the simulation grid and the computing resources at hand. Considering the time critical nature of the need to respond to these incidents, it is all too likely that the information necessary for first responders to carry out their jobs would not be available until after the fact. Realizing this limitation, the Department of Defense has requested a large number of disbursement scenarios in major cities. In this way, they would be able to respond to a specific event by utilizing the scenario that most closely matches the incident in question.

The U.S. Army is not the only organization that is currently using CFD applications to model the spread of contaminants in an urban environment. Researches at the Center for Visual Computing and Department of Computer Science at Stony Brook University in New York for instance, have been developing a method for simulating urban dispersion of smoke particles using a Multiple Relaxation Time Lattice Boltzmann Model (MRTLBM) along with a hybrid thermal MRTLBM to model thermal effects upon the flow fields using GPU computing clusters to achieve real-time simulations [2]. Similar work is being done by other research groups [1].

An alternative to these approaches is to run a series of simulations using a processing cluster or grid-computing network to generate a set of data from which accurate predictions of the dispersion of a contaminant could be made in a timely manner. Many government and scientific institutions have begun to utilize grid computing and desktop peer-to-peer (P2P) systems in order to perform parallel asynchronous iterative computations. NASA has developed a computing grid known as the Information Power Grid (IPG) to perform simulations of the aerodynamics of the next generation space shuttle and other applications [3]. The U.S. Department of Energy (DOE) has also established the Science Grid for researchers in such areas as high energy and nuclear physics [3]. In addition to these, a number of desktop P2P systems, such as SETI@HOME have been created to perform calculations using a wide range of computer platforms [4].

Utilizing a commercial computing grid, it is possible to run simulations of multiple dispersion situations on parallel systems in order to quickly develop data sets for a wide range of scenarios. With this data, it is possible to design an easy to use web-based application that allows the user to select a location from which a contaminant has been released and obtain accurate results in less time than it would take to run the simulation. In this way, not only can the issue of accurate dispersion scenario simulations be addressed accurately and quickly, it would also allow emergency first responders to immediately access this informa-

tion via a laptop computer and a wireless internet connection from close proximity to the actual contaminant release location.

In this study a working prototype of a system described above was developed which enables one to quickly analyze various aerosol dispersion scenarios based on the pre-compiled remotely stored data sets. The following sections address various issues involving distributed simulations, data storage and retrieval.

2 Method

The main idea of the approach of this study is to achieve a faster response of the risk assessment system by replacing physically-based numerical solvers with a faster information retrieval from a pre-compiled database.

Physically based solvers, such as CFD solvers, have become versatile tools which enable one to model a great variety of fluid dynamics including particle transport and deposition in complex 3D geometries. However, the accurate predictions come at a cost of long execution times. These long execution times are the penalty for the versatility of the software. At the same time, very often one does not need a geometric versatility and generality when it comes to analysis of objects and landscapes that experience no change. A city landscape, or a man made object, such as a cruise ship, are just a few examples. If one can design a simulation tool that could make predictions for a particular object only, but in a timely manner, it would be far more useful in emergency situations than a likewise accurate but slow general purpose solver.

A version of such simulation tool has been developed in this study and implemented in a prototype risk assessment system for the analysis of aerosol release and spreading in urban environments. The system consists of a simulator and a front-end interface. The simulator incorporates a physically based model of aerosol transport and is executed on high-performance computing facilities, such as computer clusters or grid environments. It collects the data of possible aerosol release scenarios. The front-end interface runs as a web-based application and retrieves the data for particular scenarios and their outcomes.

2.1 Conceptual Model

A comprehensive answer to the problem of aerosol dispersion can be provided by either conducting experiments or computer simulations by means of CFD. Both may require a long time to complete for a realistic city landscape. The outcome is usually given as three dimensional time dependent probability density functions of aerosol distribution. However, from a practical perspective the important question is not the level of contamination at each location, but rather to what degree each particular object is affected. Since the number of identified objects can be significantly less than the total number of discrete locations, this formulation can lead to considerable reduction of the size of data. As a consequence, it will result in an increase in the speed of the data retrieval process, which is an important factor for express risk assessment, especially when the database is accessed remotely.

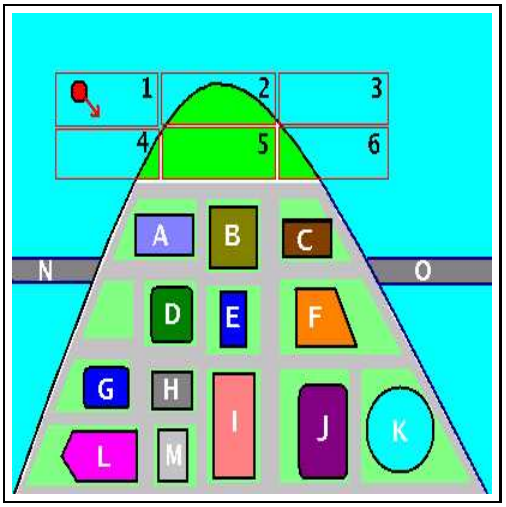

Figure 1. Aerosol dispersion in a city: 1..6 - available aerosol release data; A-O - affected objects.

Figure 1 shows a generic city map where different objects of importance, such as buildings and bridges are marked by letters (A,B,C...). The results of aerosol simulations are kept in a remote database of hit counts on all objects, corresponding to all possible release scenarios. Since the whole database is too large to be transfered over the network at once, the data are split into chunks, or *datasets*, corresponding to limited areas, marked by numbered squares (1,2,...) in Fig. 1. The size of each data set is selected so as to minimize the time of data Transfer, as well as to provide a representative set of release scenarios in a given location (see Sec.3). For each particular scenario to be investigated, the corresponding data set is retrieved, including all of the scenarios from the local area. Thus, multiple scenarios from the same location can be investigated without the need of subsequent data transfers.

Data corresponding to each release scenario contain the degree of fallout on every object in the domain of interest (i.e. downtown area). In this format the size of the data set is much smaller than it would be if three-dimensional probability density functions were used.

2.2 Physical Modeling

A simple aerosol transport model can be built from the equation of particle motion, expressed in terms of particle velocity, $\mathbf{v}(\mathbf{x}, t)$, in a given mean flow field, $\mathbf{u}(\mathbf{x}, t)$:

$$\frac{d\mathbf{v}}{dt} = C_D(\mathbf{u} - \mathbf{v}) + C_T\mathbf{u}' - \mathbf{g} \qquad (1)$$

where C_D is the drag coefficient, C_T the turbulent dif-

fusivity, $\mathbf{u}'$ is the instantaneous turbulent fluctuation vector, and $\mathbf{g}$ is the gravity acceleration vector [5].

The position of the particle at each time step is computed using the midpoint time-stepping scheme:

$$\mathbf{x} = \mathbf{x}_0 + \frac{\mathbf{v} + \mathbf{v}_0}{2} dt$$

where $(\mathbf{x}_0, \mathbf{v}_0)$ and $(\mathbf{x}, \mathbf{v})$ are the old and new coordinates and velocities of a particle. The effects of turbulent dispersion on the particles encapsulated in the second term of (1) can be accounted for by an appropriate sub-grid scale turbulence model. In particular, the RFG technique developed earlier by the authors enables one to simulate realistic effects of turbulence on the aerosols and bubbles [6]. One advantage of this approach is that the effects of turbulence can be completely decoupled from the flow solver. Thus, in a first approximation the effect of turbulent dispersion can be simulated by applying turbulent fluctuations to aerosol particles, which is superimposed on the flow velocity field, $\mathbf{u}$, which is set constant inside the domain.

Each aerosol particle is convected in a velocity field $\mathbf{u}$, and is traced inside the computational domain until it crosses the domain boundary or hits the object inside the domain. In the second event the hit count for that object is incremented. The geometry of the landscape is given by a 3D Cartesian grid, where each cell of the grid is represented by a single integer number, identifying an object, which occupies the corresponding region of space.

2.3 Distributed Simulations

Given the large size of the grid that is required to represent a realistic urban Environment, as well as a variety of other factors such as wind strengths and direction, aerosol volatility, turbulence level, etc., a large number of aerosol release scenarios, N_s, need to be played. A distributed computing facility, such as a Beowulf cluster or a grid computing environment will be the best choice for these simulations.

The number of particles released for each realization will determine the accuracy of predictions, since the variance of the result, σ, is inversely proportional to the square root of the number of particles $n(x)$ deposited at position x:

$$\sigma(x) \approx 1/\sqrt{n(x)} \tag{2}$$

Consequently, one may expect low accuracy of predictions in the regions where only few particles have reached. Given the already large number of scenarios, the number of released particles for each scenario, N_p should be selected as a trade-off between the desired accuracy and the available computing time and resources.

If one wants to keep the statistical uncertainty in (2) below a certain level, σ_{max} for most parts of the domain, then with the N_n grid nodes the appropriate number of injected particles, N_p, can be estimated from the condition $n(x) = C\,N_p/N_n$, where factor C determines the volume fraction of the affected area to the total area of the domain. When applied to (2) this condition provides the estimate for the total number of particles needed:

$$N_p = \frac{C\,N_n}{2\sigma_{max}^2} \tag{3}$$

If the average time for computing a single trajectory, τ, is given, then the time required for computing all N_s scenarios will be:

$$t = N_p\,N_s\tau/N_c \tag{4}$$

where N_c is the number of computing nodes on a cluster. It should be noted that this problem is especially suitable for parallel simulations in cluster of grid computing environments, since it involves statistical data sampling for completely independent realizations. Therefore, no interprocess communications are necessary, and the problem belongs to a so-called *embarrassingly parallel* category.

2.4 Data Reduction and Compression

Additional consideration should be given to the storage of the resultant data sets. Since particle deposition counts are recorded for each scenario, the resultant data set will consist of N_s numbers, which with a four-byte storage for an integer number will bring the size of the data set for a single object to:

$$D_o = 4N_s \tag{5}$$

And this much data should be given for each object in the domain, i.e. building, bridge, etc. Considering all objects of interest, N_{obj}, the size of the data set will be:

$$D = N_{obj}D_o \tag{6}$$

which can become a very large number even in a simplified test case (Sec.3). One way to deal with large data sets is to arrange a fast database access to these data. However, this will require a database retrieval for each value of position coordinate and angular direction, which may involve many database accesses. Secondly, with an increase in the size of the domain the size of the database files will become prohibitively large. A distributed database on a cluster may be a way out. However, there is a cheaper and simpler solution using reduced data sets and compressed files.

The size of the data set can be reduced significantly considering that the data on angular distribution are usually sparse, that is, many wind directions carry particles away from the objects causing zero deposition counts. Figure 2 shows a typical distribution of the fallout data on a particular object as a function of angle and one of the three spatial positions (height). The dependence in two other spatial directions shows similar features. It can be seen from this figure that if one stores only positive counts and discards zero

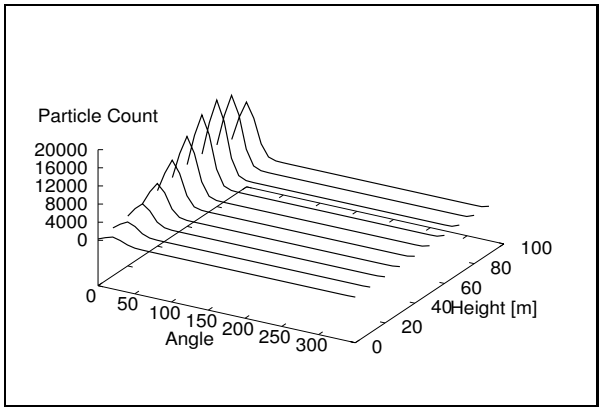

Figure 2. Typical particle count on an object for different space locations and angles.

counts, the size of the data set can be reduced. On the average, a reduction by about a factor of 4 is possible. Another size reduction can be achieved by using data compression. Standard data compression methods, such as LZW, LZ77, and others can easily reduce the size of the file by a factor of 3 or more. Thus, the size of the data set can be reduced to:

$$D_r \approx \frac{D}{4 \cdot 3} = \frac{N_{obj} D_o}{12} \tag{7}$$

2.5 Data Retrieval and Visualization

From the perspective of express risk analysis a preferred situation would be a possibility of direct access to the simulation data on a local workstation, laptop, or via the Internet. In addition to this the application should be platform independent, and should run equally well on different operation systems. A simple solution will be an efficient web interface to aerosol dispersion data. This interface should provide a reasonably short access time to the data no matter how large the total data set can be, and at the same time it should be simple and intuitive to use. Java technology provides an appropriate environment for the development of such interface. Java applications are platform independent, they can be used to access local or remote data sets, provide intuitive user interfaces and capable of 3D graphics, and can run as local or web-based applications (applets). Thus Java is a a natural choice for data retrieval and visualization.

3 Results

3.1 Simulations

A generic city landscape based upon the Pittsburgh downtown area was constructed using the voxel-based 3D graphics system [7, 8]. The whole domain was discretized on the grid with spacial resolution of $N_x = 92$, $N_y = 92$, $N_z = 32$, which corresponds to the total of $N_n = N_x N_y N_z = 270848$ nodes or possible particle release points. The landscape was populated with characteristic features like rivers, hills, bridges, park area, pavements and buildings (Fig.1). To examine all possible scenarios of aerosol release, the injection of airborne particles was simulated at each grid node with a total number of $N_a = 36$ aerosol release scenarios for each node, which corresponds to 36 wind directions spanning the whole circle of 360^o. Thus, to exhaustively cover all the possible scenarios for this particular domain size and angular resolution the total of $N_s = N_n N_a = 9750528$, or about ten million realizations of positions and wind directions has to be analyzed on particle dispersion.

To keep the average statistical error close to 10%, and estimating the typical area coverage coefficient in (3) as $C \approx 0.01$, we arrive at an estimate of required particle count as: $N_p \approx 10^5$.

On a single $1GHz$ node of a Beowulf cluster the computation of one particle trajectory with our scheme in a given landscape takes on the average $\tau = 4.5 \times 10^{-5}\ sec.$ Thus computing all the $N_s = 9750528$ scenarios with $Np = 10^5$ particles will require according to (4): $t = N_p N_s \tau \approx 5000$ days, or 14 years of computing time. If one uses a computer cluster with 100 nodes this time can be reduced to under 50 days of simulations, which will still be prohibitively large.

In addition to this, according to (5) the size of the data for a single object will be also too large for efficient Internet transfer, and close to: $D_o = 4N_s \approx 40MB$. The size of the whole data set as determined by (6) can easily approach $D \approx 1GB$, which makes it also impractical for Internet access. Using data reduction technique described in Sec.2.4 (See (7)), the size of this data set can be reduced to $D_r = D/4/3 = N_{obj} D_o/4/3 \approx 20 * 40/4/3 \approx 67MB$. This is a considerable reduction, but still not quite enough to achieve reasonably short transfer times.

3.2 Data Reduction

A faster Internet access to this information can be achieved if the data set is split among several data files, one for each object. In our case with $N_{obj} = 20$ objects this will reduce the data size for each object to: $D_{or} = D_r/N_{obj} \approx 3.3MB$.

An even greater reduction of retrievable data size can be achieved if the data for the whole domain is split into chunks or sub-domains. Such sub-division makes sense if one considers that usually the express information on the contaminant's release is needed for a specific area where the release has occurred. From this perspective, the size of the sub-domains can be selected so as to minimize the file transfer time, as well as to provide access to a representative sample of data.

To support a reasonably fast web access it was decided to maintain the size of each data file within $F \approx 100KB$. It was found by experimentation that the data file

of this size approximately corresponds to the sample size of around $N_i = 10^3$ injection points. Strictly speaking, the size of data sample F for a fixed number of injection points, N_i may vary, since the data arrays with deposition counts for different angles are represented by variable size lists to conserve space as discussed above. However, this variation is not significant. Thus, by subdividing the domain into chunks of $N_i = 10^3$ nodes we obtain the total of $N_{chunks} = N_n/N_i \approx 270$ sub-domains. A complete simulation of all possible scenarios with $N_p = 10^5$ particles in each realization for one such sub-domain will take about:

$$t_{chunk} = N_i \, Na \, Np\tau \approx 45 \; hrs$$

on a single cluster node. The computation of the whole domain on a 100-node cluster will take $N_{chunks} t_{chunk}/N_{cluster} \approx 122 \; hrs$ of cluster time. This estimate presumes that several sub-domains can be executed on a single processing node.

In the current study we used $N_{chunks} = 12$ sub-domains for possible aerosol release locations, with each sub domain containing $N_i = 1000$ aerosol release locations. In parallel simulations one sub-domain was assigned per computing node. The hit count data for each sub-domain were split between N_{obj} files, each corresponding to one object in the domain. With the number of objects $N_{obj} = 20$ this resulted in 240 files, which were compressed with a standard gzip compressor and stored on a remote server.

3.3 Implementation

The system was implemented in two main parts: distributed simulator and remote interface.

The aerosol transport and dispersion solver was implemented in C++ language and executed on a Beowulf cluster with 1GB, 1GHz computing nodes. The simulations were performed for different number of particles released per realization: $10^2, 10^3, 10^4$, and 10^5. Thus, the total of 4 runs were conducted. Total processor time required per each run was 3, 27, 266, and 2695 min for number of particles 10^2, 10^3, 10^4, and 10^5 respectively.

To demonstrate the viability of this method a prototype web interface was developed, which enabled us to test different scenarios of aerosol release and dispersion (mulphys.com/sim). The interface is written in Java language and provides a 3D representation of a city landscape with the possibility of navigating through the landscape, arbitrary positioning of the aerosol source, and setting wind direction (Fig.3). The applet also performs a real time simulation of aerosol propagation and dispersion for a limited number of particles.

The count of particles deposited on the selected object is constantly updated as the simulation is progressing. At the same time, the original count of particles retrieved from the cluster simulation data is displayed alongside the

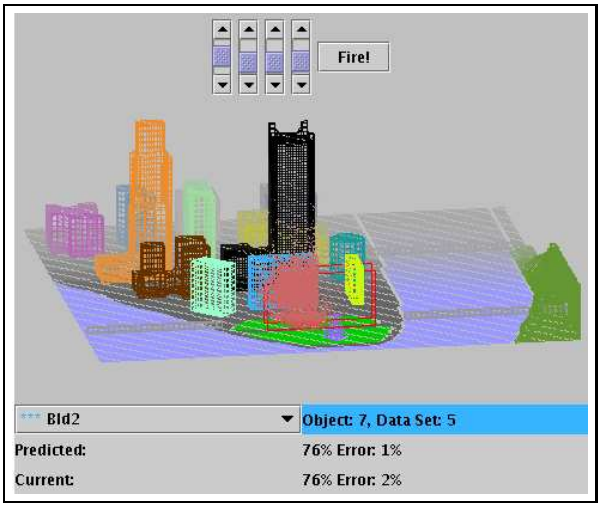

Figure 3. Web interface to simulate aerosol release in a city

real-time simulation count. For the purposes of better interpretation, the count of particles, n, for each object is converted into the deposition probability by dividing it by the total number of particles released, N. The statistical error of the results is shown alongside with the probability count as being proportional to $n^{-1/2}$, according to (2).

The simulation in the applet is performed with a limited number of particles, and serves the purpose of comparing its execution time with a fast retrieval of the pre-assembled simulation data. The accuracy of the applet results is also less than that obtained from the data, since the latter were performed with a much higher number of particles.

When tested on a 1GB 1HG workstation with 100MB Internet connection the speed of retrieval of each dataset was on the average 10 to 20 times faster than the Java execution of the simulation for 10^4 particles. Considering that the retrieved data set consisted of 1000 realizations, it leads to a much greater speedup if multiple local scenarios need to be analyzed.

4 Conclusions

The method of this study facilitates express risk analysis of multiple scenarios for complex physical events. It is based on information retrieval from the compressed sets of data obtained in prior exhaustive simulations. The method was applied to a problem of express analysis of aerosol contamination in the city. The results show that this approach is viable and provides an efficient tool for fast analysis of different contamination scenarios.

One characteristic feature of the current approach is the reliance on completely decoupled simulations for generating exhaustive scenarios for the whole parameter space. In this way no communication is required between the computing nodes of the grid and the computations fall in the category of the so-called *embarrassingly parallel* simula-

tions. This is a very favorable feature of the method since it makes it ideally suitable for massively parallel computer architectures, such as grid computing environments. In fact, the loosely coupled nature of grid computing architectures makes them unsuitable for many scientific and engineering problems. And this study presents a rare example when grid computing appears to be the best suited computing platform for the task.

Simulations of aerosol propagation in a generic city landscape were conducted on a Beowulf cluster and sampled data sets were obtained. After the appropriate reduction and compression of these data sets it was possible to demonstrate the feasibility of fast retrieval of these data via the Internet for express analysis of different scenarios. A prototype web interface for express simulations of particle deposition scenarios was developed and tested (mulphys.mae.wvu.edu/sim/demo).

Acknowledgments

The authors would like to acknowledge West Virginia University Research Corporation for sponsoring this work.

References

[1] J. Thilmen. Harm's way. *Mechanical Engineering*, 127:22–24, 2005. bioterror CFD.

[2] Feng Qiu, Ye Zhao, Zhe Fan, Xiaoming Wei, Haik Lorenz, Jianning Wang, Suzanne Yoakum-Stover, Arie Kaufman, and Klaus Mueller. Dispersion simulation and visualization for urban security. In *VIS '04: Proceedings of the conference on Visualization '04*, pages 553–560, Washington, DC, USA, 2004. IEEE Computer Society.

[3] William E. Johnston. Using computing and data grids for large-scale science and engineering. *Int. J. High Perform. Comput. Appl.*, 15(3):223–242, 2001.

[4] James C. Browne, Madulika Yalamanchi, Kevin Kane, and Karthikeyan Sankaralingam. General parallel computations on desktop grid and p2p systems. In *LCR '04: Proceedings of the 7th workshop on Workshop on languages, compilers, and run-time support for scalable systems*, pages 1–8, New York, NY, USA, 2004. ACM Press.

[5] C. Crowe, M. Sommerfeld, and Y. Tsuji. *Multiphase Flows with Droplets and Particles*. CRC Press, 1998.

[6] A. Smirnov, S. Shi, and I. Celik. Random flow generation technique for large eddy simulations and particle-dynamics modeling. *Trans. ASME. Journal of Fluids Engineering*, 123:359–371, 2001.

[7] A.V. Smirnov, H. Zhang, and B. Sowers. Voxel-based volume graphics system for multi-physics modeling. In *The 8th World Multi-Conference on Systemics, Cybernetics and Informatics*, volume 5 of *Computer Science and Engineering*, pages 144–149. International Institute of Informatics and Systemics, 2004.

[8] A.V. Smirnov, H. Zhang, A. Burt, and I. Celik. Fuel-cell simulator interface. *Journal of Power Sources*, 138/1-2:187–193, 2004.