**INTRODUCTION**

Particles from emission and other processes undergo physical events, such as coagulation. The atmospheric consequences of these events are of special interest, as they may contribute to climate change, or negatively impact human pulmonary health. Because of the inherent difficulty in collecting real-world data for these processes, they are often simulated using computer models. These models are typically executed using many thousands of particles. Typical implementations suffer from one of the following:

1. Computation time, which scales with the number of particles, becomes prohibitively slow on personal computers.
2. It is expensive to alleviate this performance issue with supercomputers.

GPU computing offers an affordable alternative to supercomputers for massively parallel applications. Our main contribution is the use of CUDA to speed up routines for the following in existing Particle-resolved Monte Carlo (PartMC) code for atmospheric aerosol simulation:

1. Condensation of water vapor on aerosol particles
2. Aerosol particle coagulation

**CONDENSATION**

Particle condensation is the accumulation of water vapor on aerosol particles. This ultimately affects the particle wet diameter. As humidity increases, the amount of water condensing on particles increases and the diameter of each particle expands. Condensation is a contributing factor of cloud formation. The PartMC condensation code simulates the change in diameter of particles due to condensation. Since large numbers of particles are needed to get useful results it is important that this code run faster and more efficiently. Parallelizing this condensation code was very straightforward. The set of wet diameters is governed by a system of ODE’s (given in [1]), and the method the ODE solver uses (a backwards difference method) requires expensive, independent Newton iterations for each particle. Since each particle does not communicate with any other particle, the CUDA parallelization of these routines was trivial.

**COAGULATION**

Particle coagulation is the process whereby aerosol particles collide and stick together, effectively forming a new, larger particle. Any two particles have a certain probability per unit time of coagulating. This probability is determined by the value of a kernel type, which depends on properties of the two colliding particles. Below is a plot of the Brownian kernel, which gives a fairly good approximation for what actually occurs in nature.

![Figure 1: The kernel takes on larger values for particles with very different volumes. For simplicity, all the particles in the plot have the same density.](image)

The kernel is a function from $R^3 \rightarrow R$, as it depends on the volumes and densities of two separate particles.

**MANTAINING SETS OF COAGULATING PARTICLES**

In a single time step, one particle may coagulate with multiple other particles (usually a large particle with several smaller ones – this is more likely, as evident in Figure 1). Keeping track of these sets is essentially the same problem as the union-find structure is most appropriate (though in practice, we use a slightly more complicated technique).

**CONSERVATION RESULTS**

![Figure 2: Determining coagulations in parallel. After pairwise coagulations are determined from kernel values computed on the graphics card, a union-find algorithm (performed on the GPU) maintains connected components in a forest of disjoint trees. These connected components are then combined.](image)

**REFERENCES**


**ACKNOWLEDGMENTS**

We gratefully acknowledge support from NSF grants 1004311 and 0934491. Additionally we acknowledge valuable mentorship and assistance from Matthew West, Jeffrey Curtis, Matthew Michalovski, Craig Zilles, and Jack Pickeum.

**PARALLEL TAU LEAPING**

Instead of performing kernel evaluations, schedule them on the GPU.

The GPU evaluates kernels and random numbers, returning a vector of booleans.

If two particles coagulate, add an edge between them in the particle graph (see figure 2).

Combine coagulated sets of particles.

**COAGULATION: SERIAL TAU LEAPING**

Suppose the simulation involves $N_p$ particles. To avoid testing all $N_p \times (N_p - 1)$ particle pairs for coagulation, PartMC stochastically samples particle pairs, using a binned accept-reject approach for efficiency purposes.

Below is pseudocode for this binned tau-leaping method for coagulation, similar to what is given in [2]:

1. Divide diameter axis into bins, on a log scale $V$ is the computational volume.
2. $N_b(b)$ is number of particles in bin $b$.
3. $\mu(b, i)$ is mass vector of the $i$-th particle in bin $b$.
4. $K_{\text{bin}}(b_1, b_2)$ is a precomputed upper bound on the kernel for any particles from bins $b_1$ and $b_2$.
5. $\Delta t$ is the timestep.
6. for all bin pairs $(b_1, b_2)$ with $b_1 \leq b_2$ do
7. $N_{\text{next}} = N_b(b_1) N_b(b_2)/(1 + \delta_{b_1, b_2})$
8. $N_{\text{next}} \leftarrow \text{Point}(K_{\text{bin}}(b_1, b_2) \Delta t N_{\text{next}} / V)$
9. for $N_{\text{next}}$ repetitions do
10. randomly choose particles $i_1$ and $i_2$ uniformly in bins $b_1$ and $b_2$
11. $\Delta_t \leftarrow K_{\text{bin}}(\mu(b_1, i_1), \mu(b_2, i_2))$
12. randomly choose $r$ uniformly in $[0, 1]$
13. if $r < K_{\text{bin}}(\mu(b_1, i_1), \mu(b_2, i_2))$ then
14. coagulate the two particles, updating the arrays $X(b)$ and $\mu(b, i)$
15. end if
16. end for
17. end for

**PARTICLE SIMULATIONS IN CUDA**

Stephen Macke and Rachel Turner
University of Tulsa, Youngstown State University
stephen-macke@utulsa.edu, rturner@ysu.edu

**CONSERVATION RESULTS**

![Figure 3: CUDA Newton iterations resulted in significant speedups.](image)

![Figure 4: Merging kernel evaluations to the GPU results in significant speedups.](image)