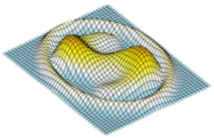
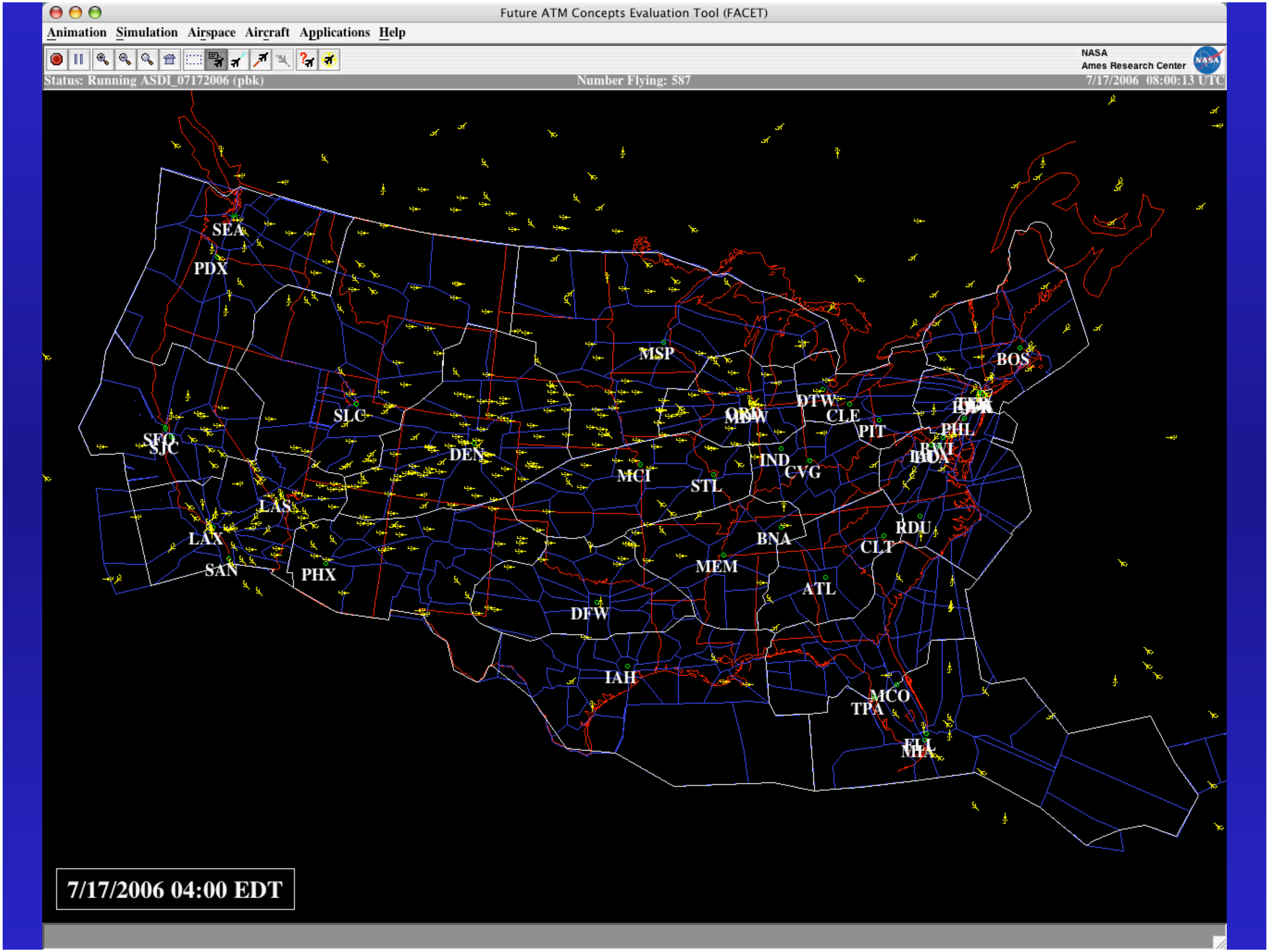


Faster Simulations of the National Airspace System

PK Menon
Monish Tandale
Sandy Wiraatmadja
Optimal Synthesis Inc.

Joseph Rios
NASA Ames Research Center

NVIDIA GPU Technology Conference 2010, San Jose, CA


Future ATM Concepts Evaluation Tool (FACET)
Animation Simulation Airspace Aircraft Applications Help
NASA
Ames Research Center
Status: Running ASDI_07172006 (pbk)
Number Flying: 587
7/17/2006 08:00:13 UTC
SEA
PDX
SFO
SJC
LAX
SAN
LAS
PHX
SLC
DEN
MSP
MCI
DFW
IAH
STL
MEM
BNA
ATL
IND
CVG
DTW
CLE
PIT
PHL
BOS
RDU
CLT
MCO
TPA
7/17/2006 04:00 EDT

Research Achievements

- Parallelized trajectory prediction code of a high-fidelity airspace simulator
- Ported parallel implementation to CUDA architecture
- Gained 240X speed-up over original implementation

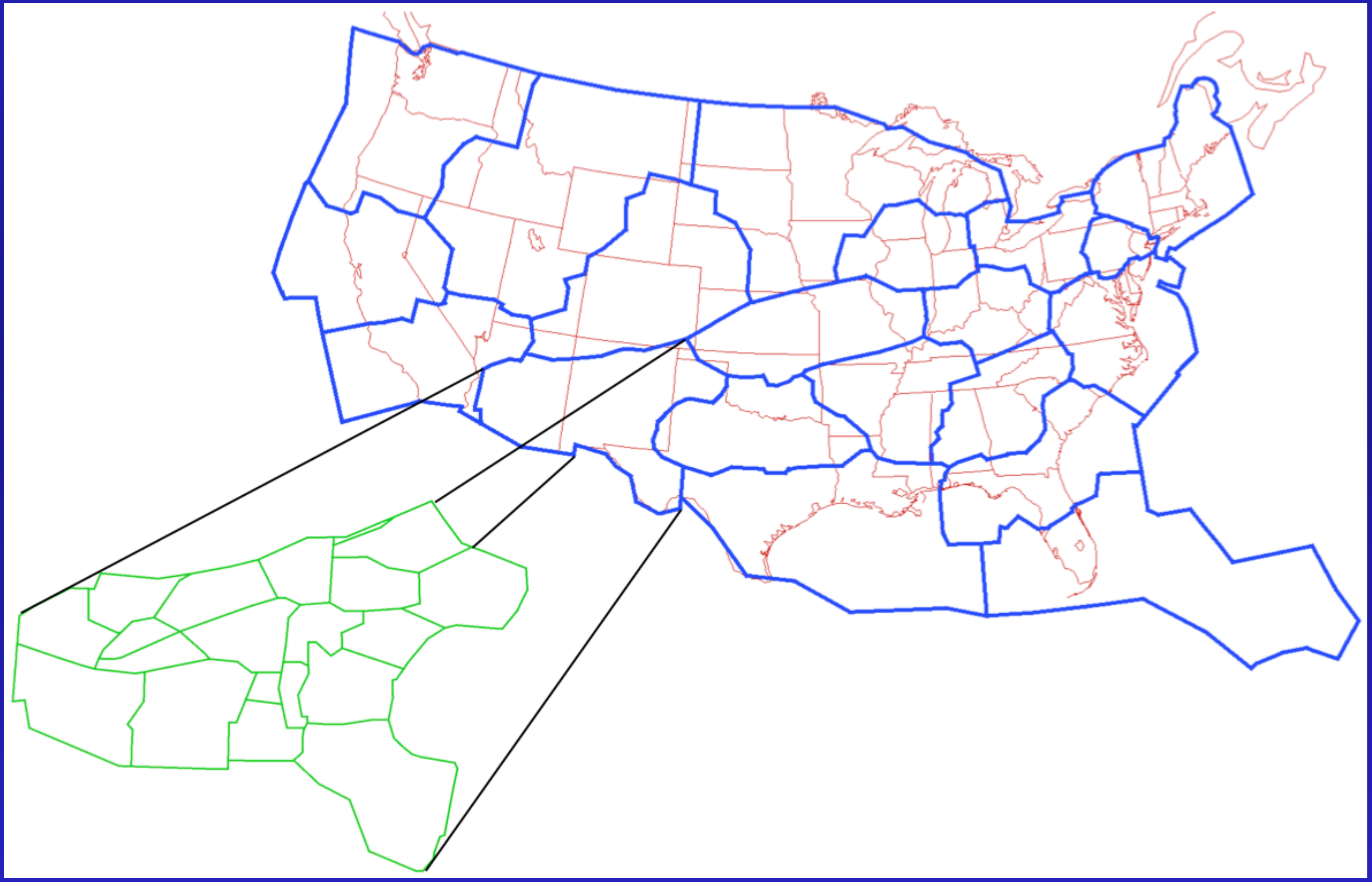
24-hour simulation with 35,000 aircraft completed under 2.5 seconds

Outline

- Background
  - Air Traffic Management overview
  - Airspace Simulations
  - Traffic Flow Management
- Accelerating simulations for Traffic Flow Management Research
- Porting code to Compute Unified Device Architecture
- Experimental results
- Concluding remarks

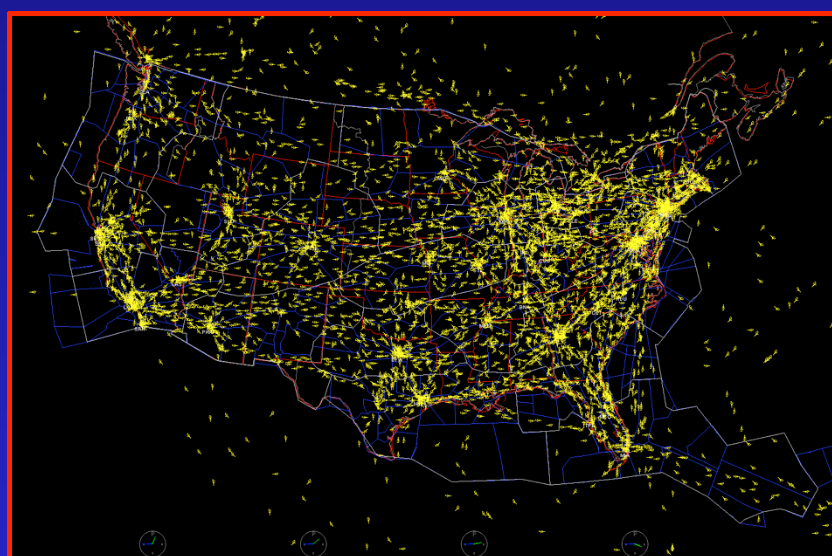
BACKGROUND

Air Traffic Management



Simulating Air Traffic

- FACET: Future ATM (Air Traffic Management) Concepts Evaluation Tool
- Java front end, C back end
- Takes traffic and weather data as input
- Works in one of two modes: Simulation or Playback
- In Simulation Mode, position of each aircraft propagated forward through calculations at each time step
- Through GUI or an API, user can query or modify state of simulation

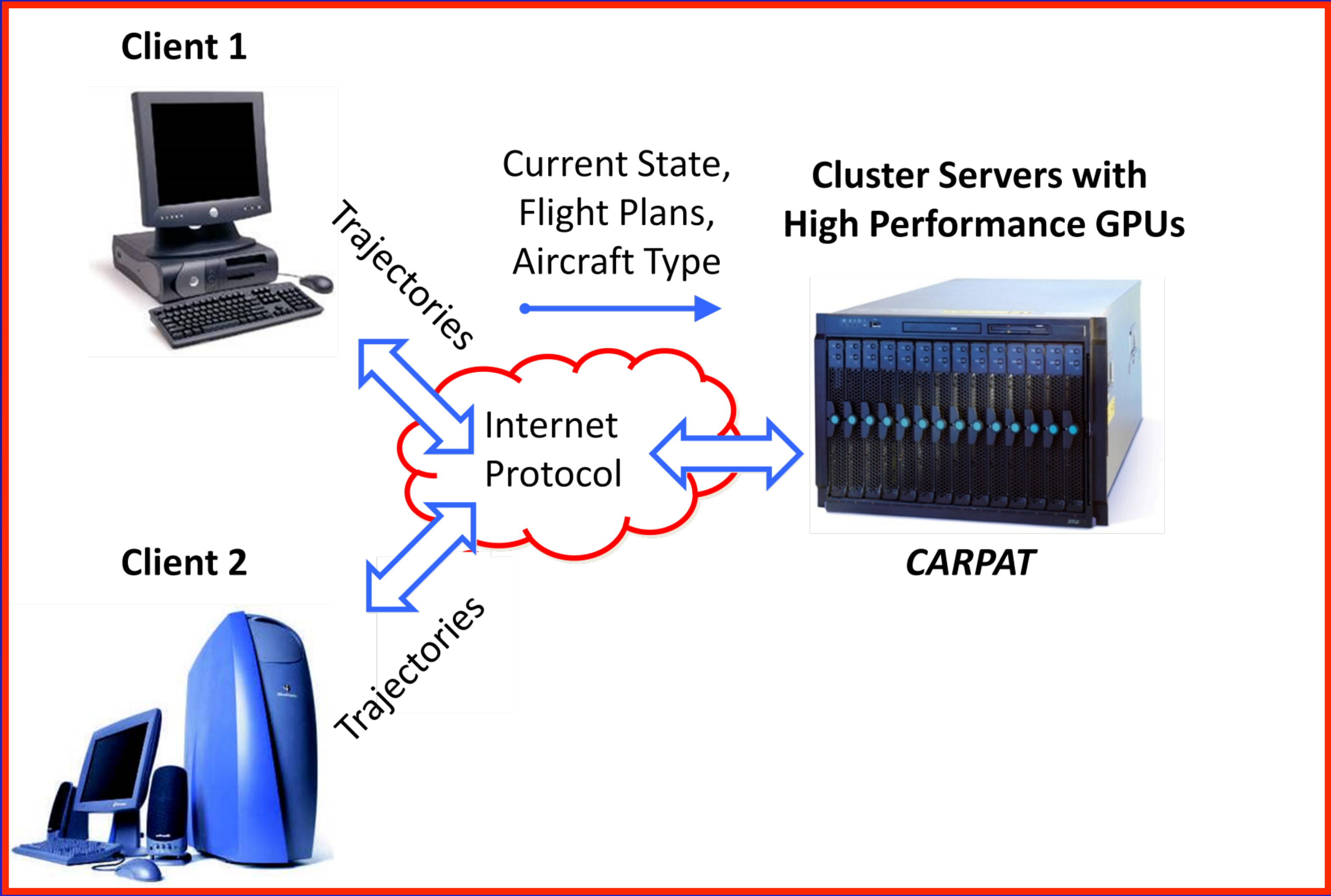


Traffic Flow Management Problem Definition

- Given a set of scheduled flights and a set of capacity values, how should those flights be held to minimize delay costs while respecting capacities?
- Search for solutions involves simulation of air traffic to build schedules and check capacity violations
- Future ATM Concepts Evaluation Tool (FACET) is powerful, commonly-used simulator

ACCELERATING SIMULATIONS FOR TRAFFIC FLOW MANAGEMENT RESEARCH

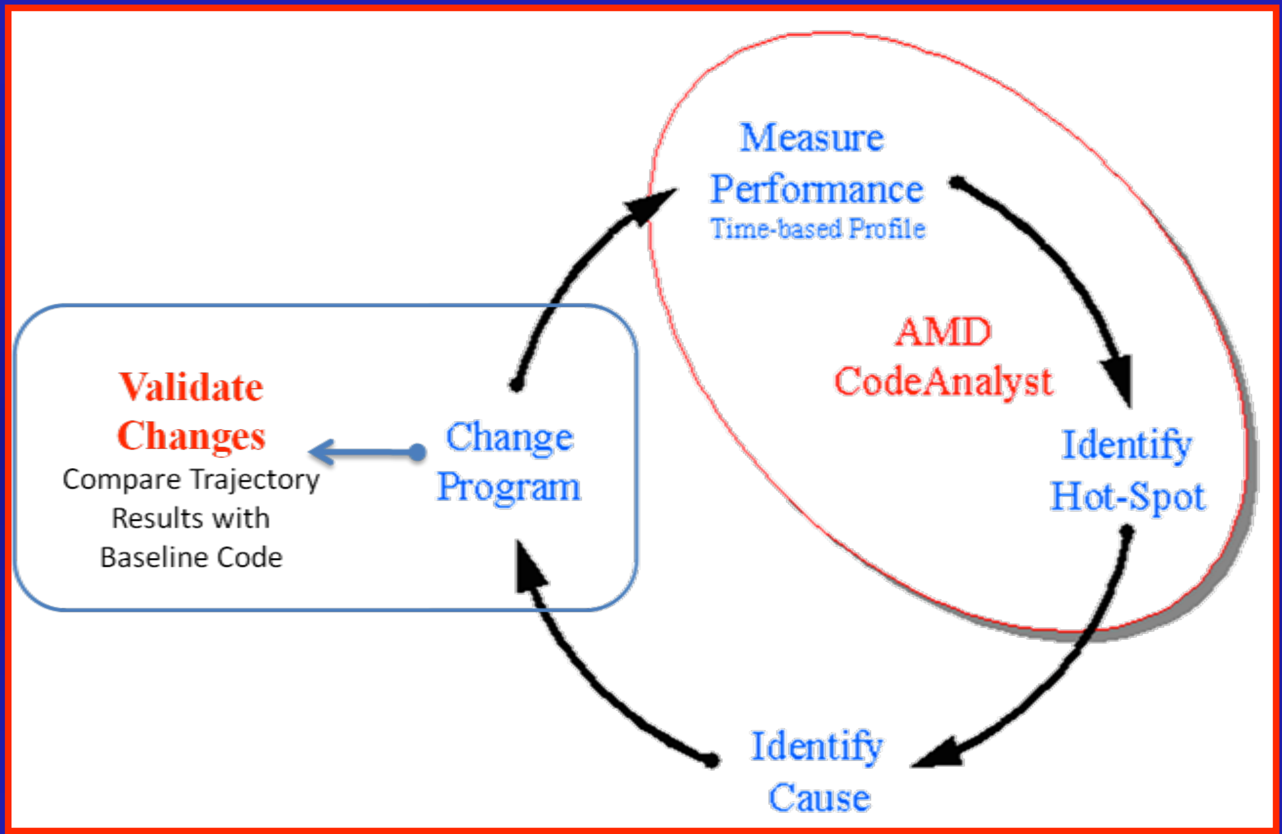
Computational Appliance for Rapid Prediction of Aircraft Trajectories (CARPAT)



Accelerating Simulations of Traffic Flow

- Software Profiling and Tuning
- Parallelization
  - Cluster Implementation
  - Multi-Core Implementation
  - GPU Implementation
    - GPU as a co-processor
    - GPU as the primary processor

Software Profiling and Tuning



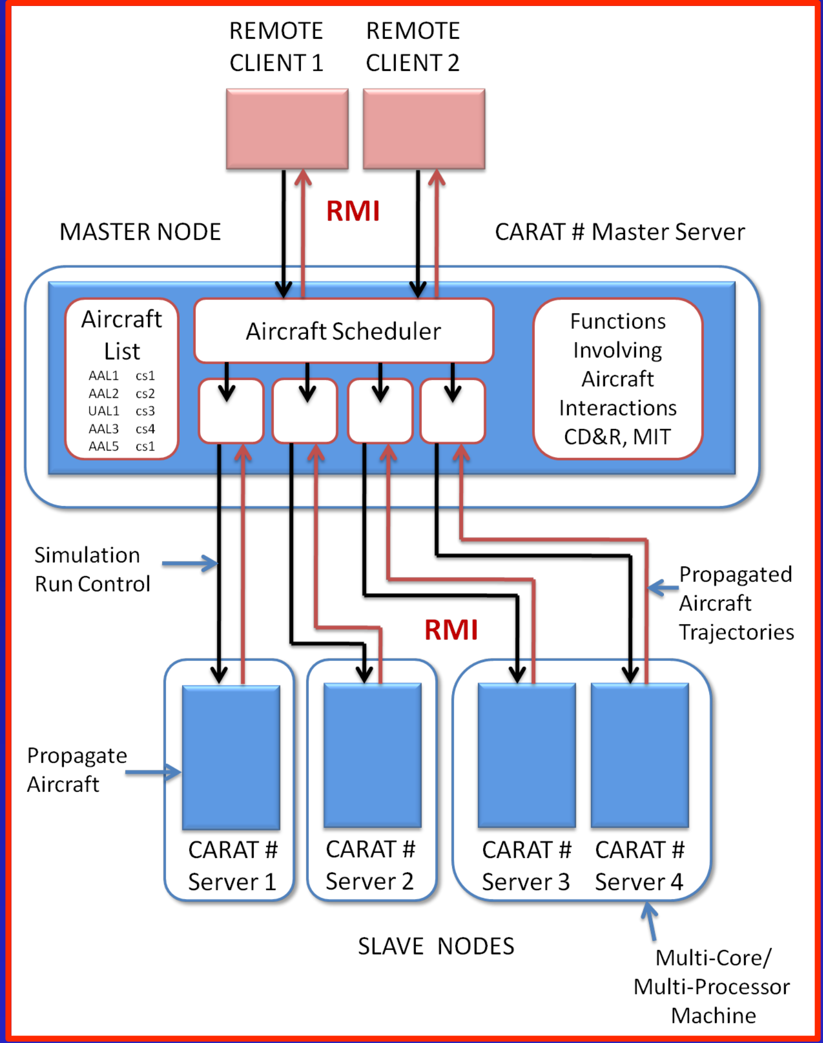
- Optimized table lookup, optimized linked list management, elimination of redundant calculations in case of no change of state.

3.1X faster

Cluster Implementation

- Distribute computational load between multiple processors connected over a high-speed network
- FACET cluster implementation using Java RMI
- Max acceleration
  - Without data assembly at Master: **4.8x** (10 slaves)
  - With data assembly at Master: **2.4x** (12 slaves)
- Acceleration with software tuning included:

5.3X faster



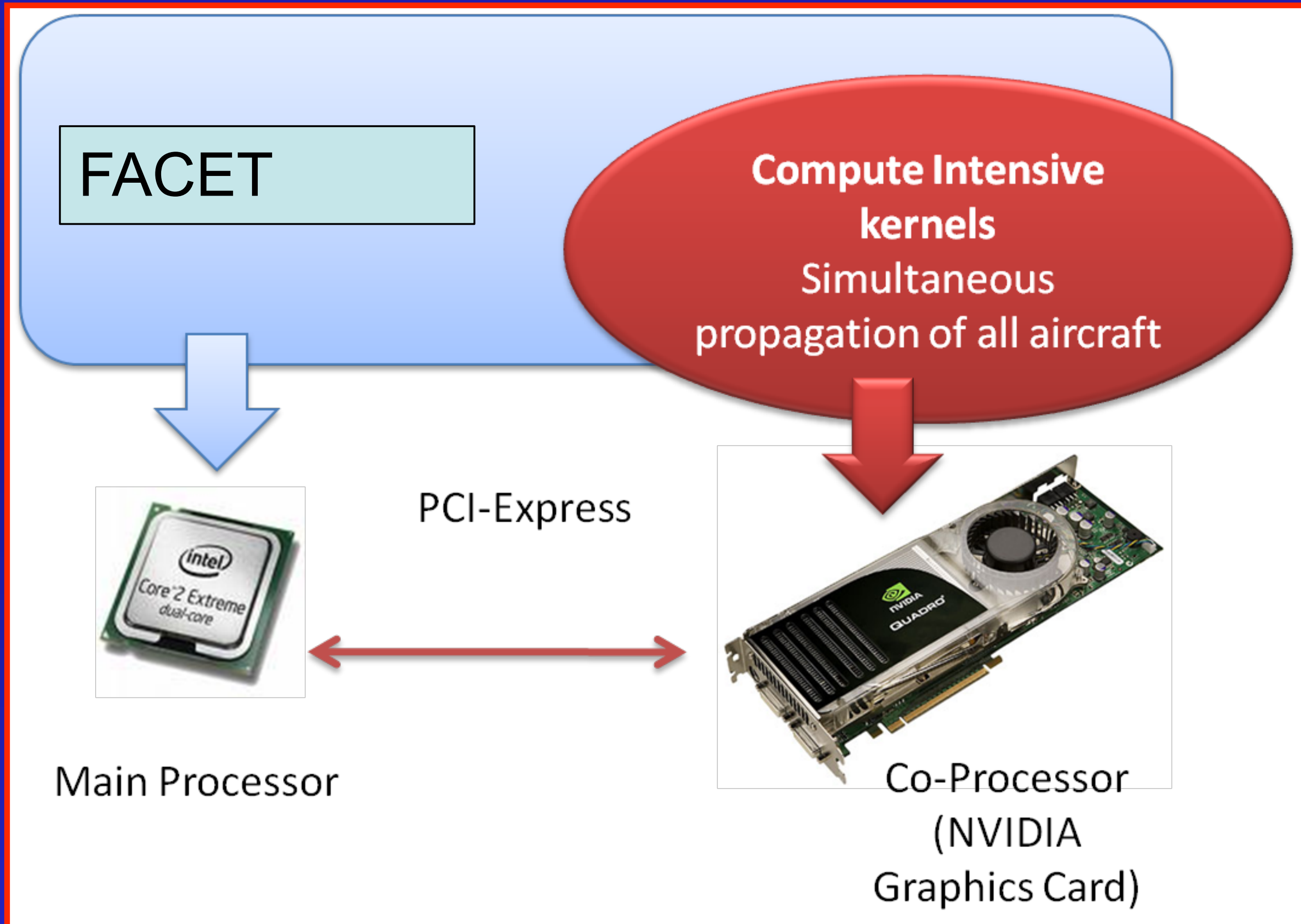
Multi-Threaded FACET Implementation

- Propagate aircraft for a single time step in parallel
- Divide aircraft list into *N* sub-lists for a *N*-core processor
- Run propagation loop over each sub-list concurrently in separate threads
- Implemented using POSIX threads

8.0X faster

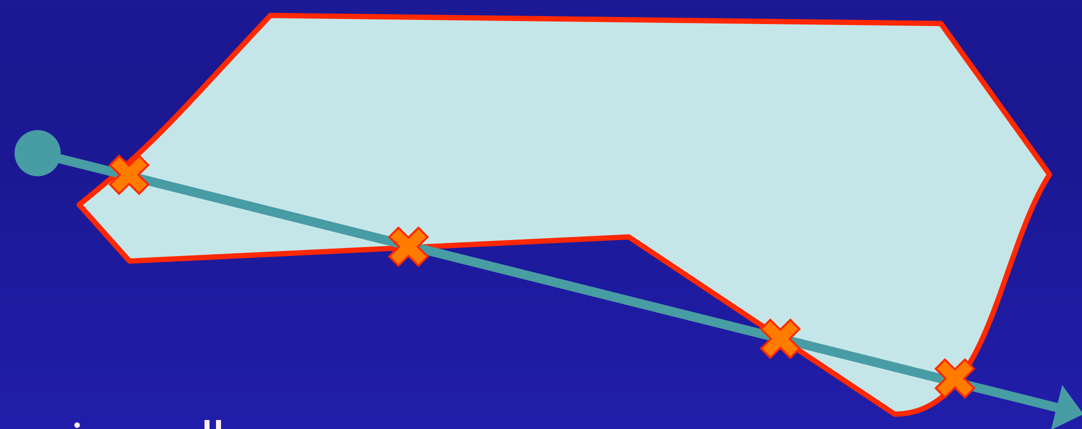
PARALLELIZING TRAFFIC FLOW SIMULATION FOR A GPU

Using the GPU as a Coprocessor



GPU Implementation of `sim_inBoundary`

- Code Profile: functions that consumed most amount of runtime
- `sim_inBoundary`: Check whether a point is inside a polygon
- Ray Casting Algorithm



- Number of edges is small
- Data access time > time reduced by parallel computation over edges
- 24-hour simulation: 26 minutes

10X slower

GPU as the Primary Processor

- Parallelization lesson learned
  - Want significant run time for single call
  - Need minimum amount of data transfer between host and GPU
- Propagation of every aircraft for single time step?

Need for a new Trajectory Predictor

- FACET: Mixed C and Java Application
- FACET: Aircraft data contained in linked lists
- Develop CARPAT Trajectory Predictor completely in C

TRAJECTORY PREDICTOR

Trajectory Predictor

- Flight data input: FACET track file

  TRACK JBU222 A320 422700 853000 498 370 93 ZAU ZAU23
  FP_ROUTE LGB./.PMM.J70.LVZ.LENDY5.JFK

- Aircraft performance data: BADA (Base of Aircraft Data)

- Airspace data such as airports, named waypoints, airways, sector boundary data, etc.

Trajectory Predictor Model

- Calculation of great-circle heading between waypoints of the flight plan

$$\psi = \tan^{-1}\left(\frac{\sin(\tau_2 - \tau_1)\cos(\lambda_2)}{\sin(\lambda_2) * \cos(\lambda_1) - \sin(\lambda_1) * \cos(\lambda_2) * \cos(\tau_2 - \tau_1)}\right)$$

- Latitude propagation

$$\lambda_2 = \lambda_1 + \frac{V_G \cos(\psi)}{(R_E + h)}\Delta t$$

- Longitude propagation

$$\tau_2 = \tau_1 + \frac{V_G \sin(\psi)}{(R_E + h)\cos(\lambda_1)}\Delta t$$

- Ground speed

$$V_G$$

- Altitude propagation

$$h_2 = h_1 + \dot{h} \cdot \Delta t$$

- Climb/descent rate

$$\dot{h}$$

Trajectory Output Data

Flight data at every 30-second interval:

- Time since the start of simulation (seconds)
- Flight mode (Preflight, Climb, Cruise, Descent, Landed)
- Latitude (degrees): -90 to +90: North positive
- Longitude (degrees): -180 to +180: East positive
- Altitude (feet)
- True air speed (knots)
- Altitude rate (feet/second)
- Heading angle (degrees)
- Flight path angle (degrees)
- Sector Index

Multicore Trajectory Predictor

- Automatically detects number of cores on the computer
- Splits aircraft into lists to run in parallel on different cores
- Runtimes for 24-hour simulation

# Threads	Propagation Time (s) w/o thread affinity	Propagation Time (s) w/ thread affinity	Propagation Speed up over single thread	Trajectory output file writing (s)
1	65.47	66.31	-	36.96
2	55.88	37.57	1.76X	24.69
3	63.41	24.76	2.22X	18.81
4	65.66	24.76	2.68X	14.91

PORTING CODE TO CUDA

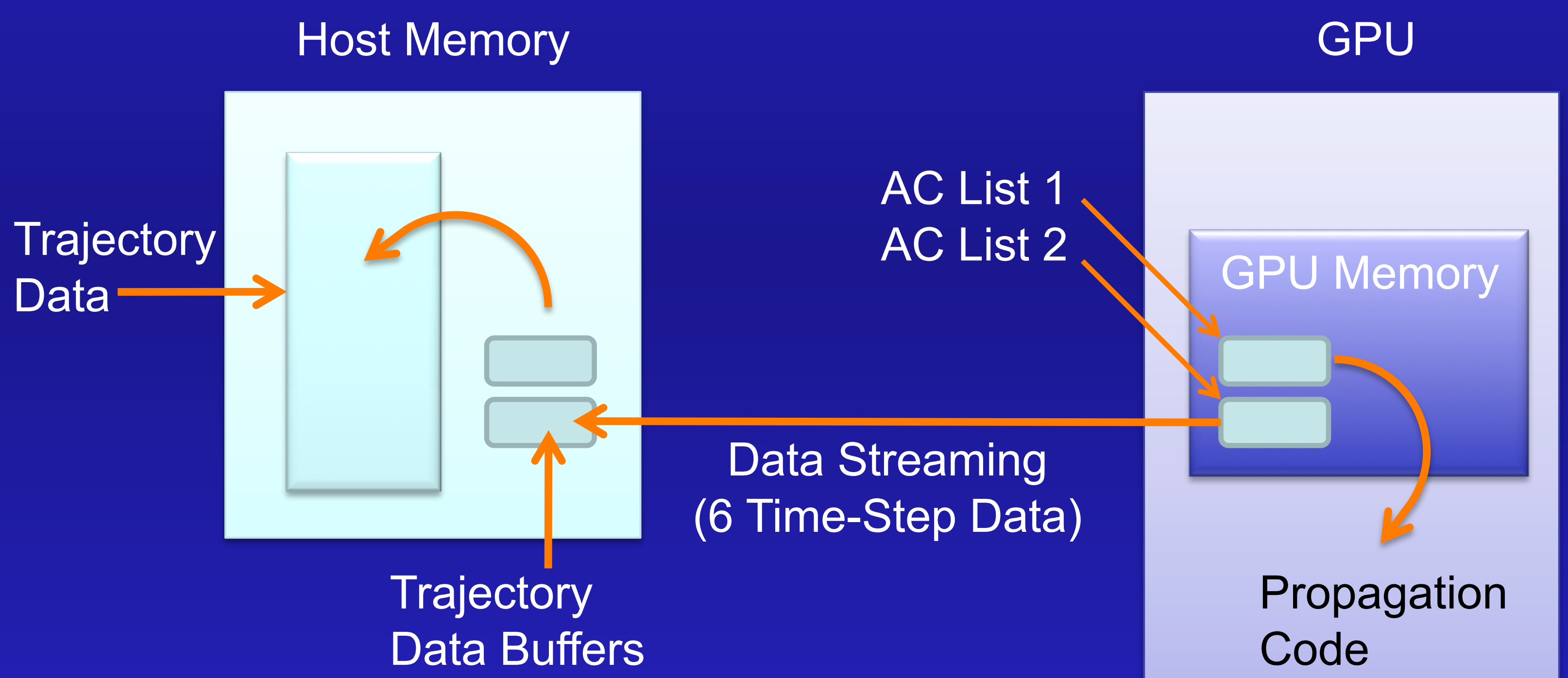
Obstacles to GPU Implementation

- Integer and Single-Precision Floating Point supported
- Dynamic memory allocation inside a structure is not allowed
- Host functions cannot be called from the device functions
- Recursive functions are not allowed
- Linked Lists cannot be used

Trajectory Predictor Data Storage

- All aircraft data is stored in global memory, because read/ write access is needed
- Must minimize high-latency memory access
- Each aircraft has a large trajectory data structure
- Use streams to hide the latency

Trajectory Data Streaming: GPU To Host



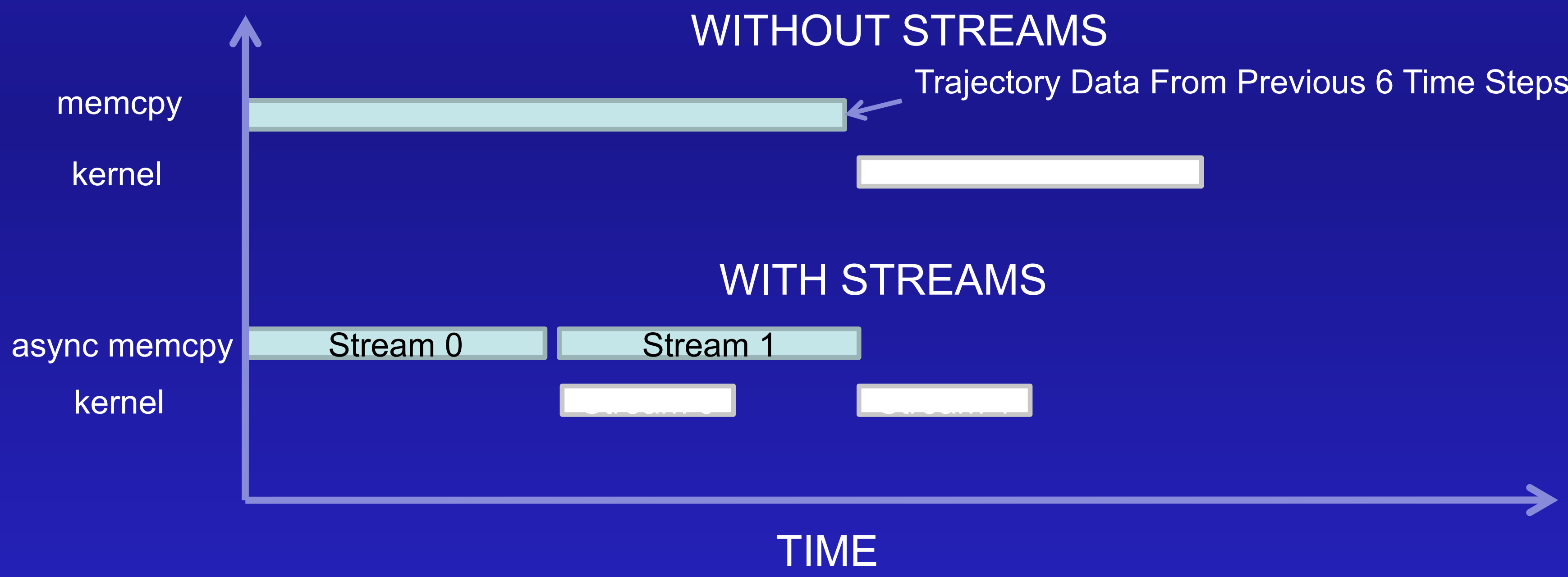
Trajectory Data Streaming: GPU To Host

- Streams
  - Memory transfer and kernel execution of each list must happen sequentially, but memory transfer and kernel execution of separate lists can happen simultaneously
  - Requires asynchronous memory transfer

- Asynchronous Memory Transfer
  - Does not block CPU computations by default
  - CPU computations be blocked using `cudaStreamSynchronize()`
  - Host memory has to reside in the page-locked (pinned) memory

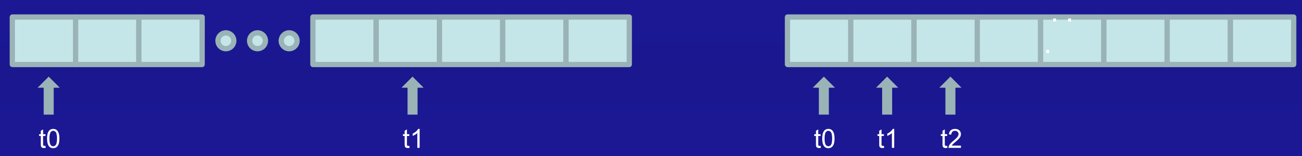
Timeline Comparison

- After the trajectory data in GPU is full (6 time-step propagations):



Memory Optimization

- Coalescing global memory access of aircraft data
  - Original implementation: array of structures
  - Modified Implementation: independent arrays for each field



Original : prevents coalescing (size of structure is large)

Modified: leads to coalescing

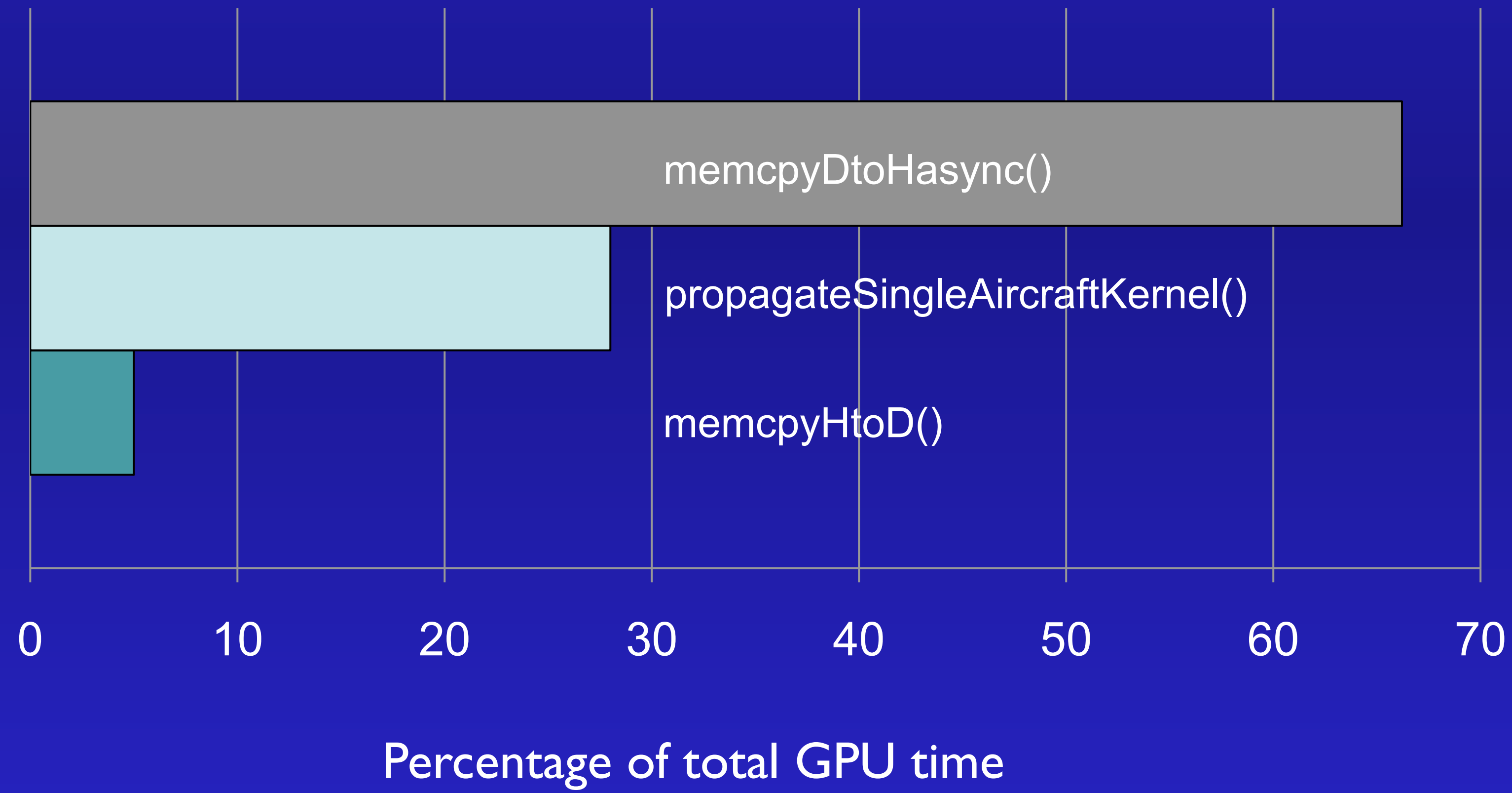
- Use registers to avoid redundant memory transfers, whenever possible
  - Registers are limited, use caution when allocating them

Optimization of Parallelization

- Increase parallelization
  - Limiting the number of registers allows bigger blocks and improves thread parallelism
- Minimize branching within a warp
  - Reducing the if-else statements and while loops whenever possible
  - Performance lookup
    - Index is based on altitude, found using a ‘while’ loop
    - Instead, altitude table has 500-foot intervals, can be calculated:

$$index = \text{floor}\left(\frac{current\_altitude}{500}\right)$$

CUDA Profile Summary



EXPERIMENTAL RESULTS

GPU Hardware

- GeForce 9800 GT (112 cores)

- TeslaC-1060 (240 cores)

GPU Comparison

	GeForce 9800GT	Tesla C 1060
# Cores	112	240
Processor clock	900 MHz	1300 MHz
Dedicated memory	0.5 GB	4 GB
Memory clock	900 MHz	800 MHz
Memory interface width	256 bit	512 bit
Memory bandwidth	57.6 GB/s	102 GB/s
Simulation time	**4.18 s**	**2.42 s**

1.7X faster

GPU Implementation Results

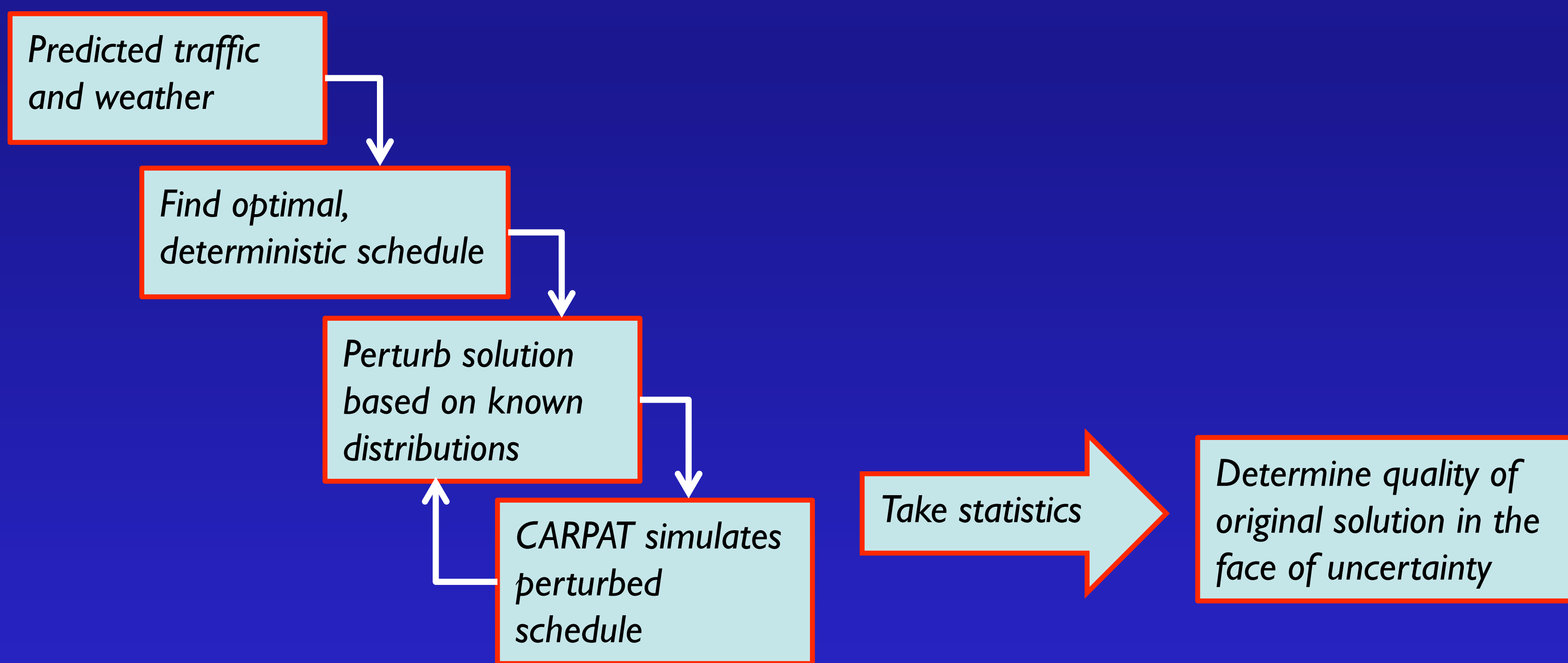
Implementation	Runtime (s)	Speed-up w.r.t. FACET	Speed-up w.r.t. CPU	Speed-up w.r.t. Multi-threaded
Original FACET	586.9			
Trajectory Predictor (CPU)	46.6	12.6X		
Trajectory Predictor (Multi-threaded)	24.5	23.9X	1.9X	
Trajectory Predictor (GPU)	2.4	242.5X	19.2X	10.1X

243X faster

POTENTIAL USES

Monte Carlo Simulations

- There is uncertainty in traffic flow management
- Many models for traffic flow management are deterministic
- Many uncertainties have been quantified in previous research



Applications in Other Domains

- Traffic flow analysis of other transportation systems
  - Automobile
  - Train
  - Space
- Job-shop scheduling

CONCLUDING REMARKS

Research Achievements

- Parallelized trajectory prediction code of a high-fidelity airspace simulator
- Ported parallel implementation to CUDA architecture
- Gained 240X speed-up over original implementation

24-hour simulation with 35,000 aircraft completed under 2.5 seconds

Questions?

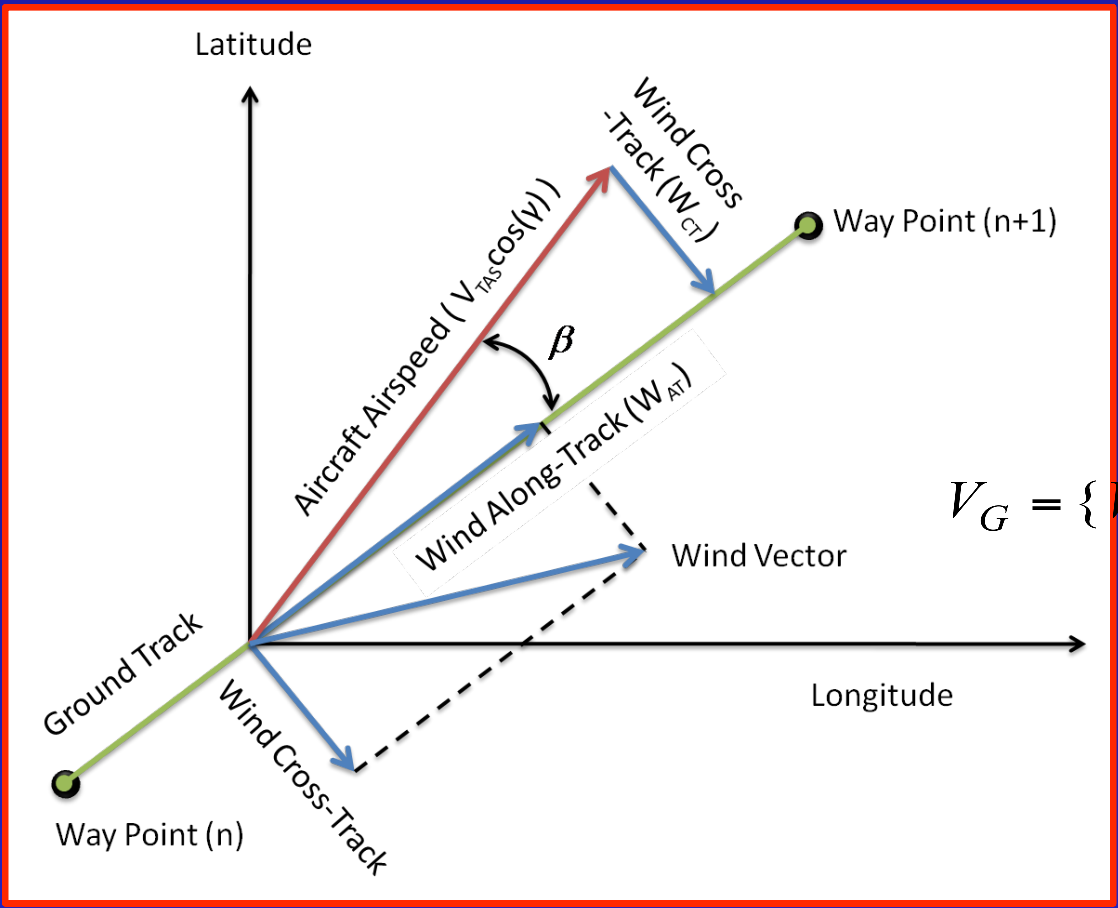
Joseph.L.Rios@nasa.gov
NASA Ames Research Center

P. K. Menon, Monish D. Tandale & Sandy Wiraatmadja
Optimal Synthesis Inc.

BACKUP SLIDES

Effect of Wind Along Track

- Reading True North and True East Wind Components from the RUC File



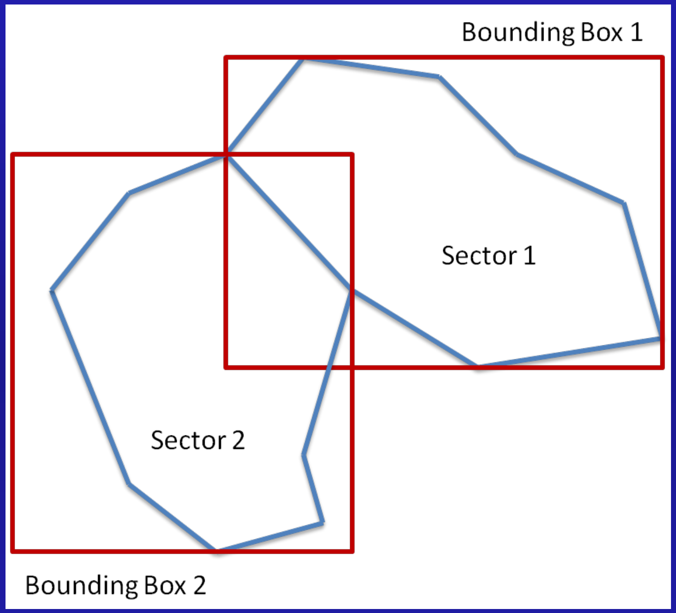
$$\beta = \sin^{-1}\left(\frac{W_{CT}}{V_{TAS}\cos(\gamma)}\right)$$

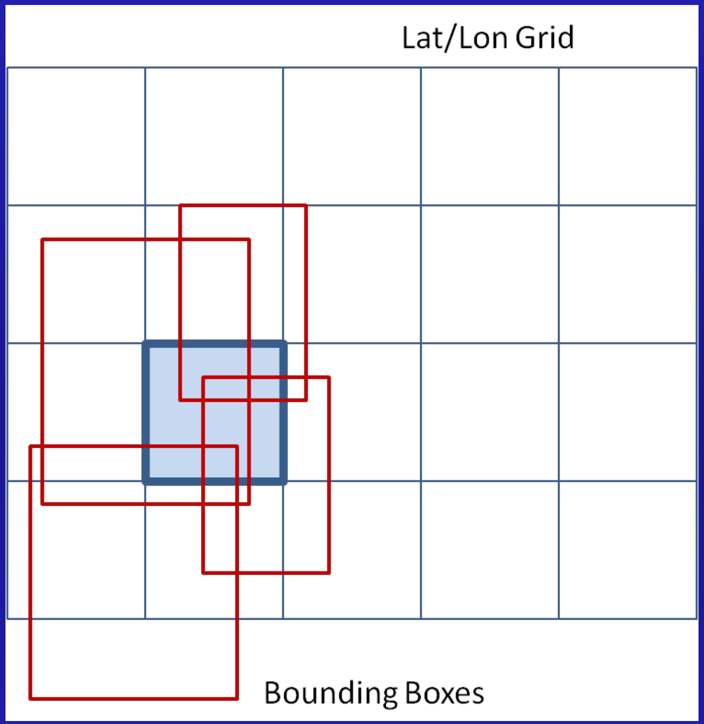
$$V_G = \{V_{TAS}\cos(\gamma)\}\cos(\beta) + W_{AT}$$

Incorporating RUC Weather Data

- RUC: Rapid Update Cycle: NOAA/NCEP weather forecast
  - 13 km resolution, 50 Vertical Levels
- Wind Data: u & v components of the wind (Lambert Conformal Projection)
  - At a given 13km grid cell for a geopotential height
- Convert to True North and True East Wind Components at a given (Lat, Lon, Alt)
- Script to Download and Automatically Store RUC Data Files from the NOAA Website

Identifying the Current Sector for every Aircraft at every Time Step



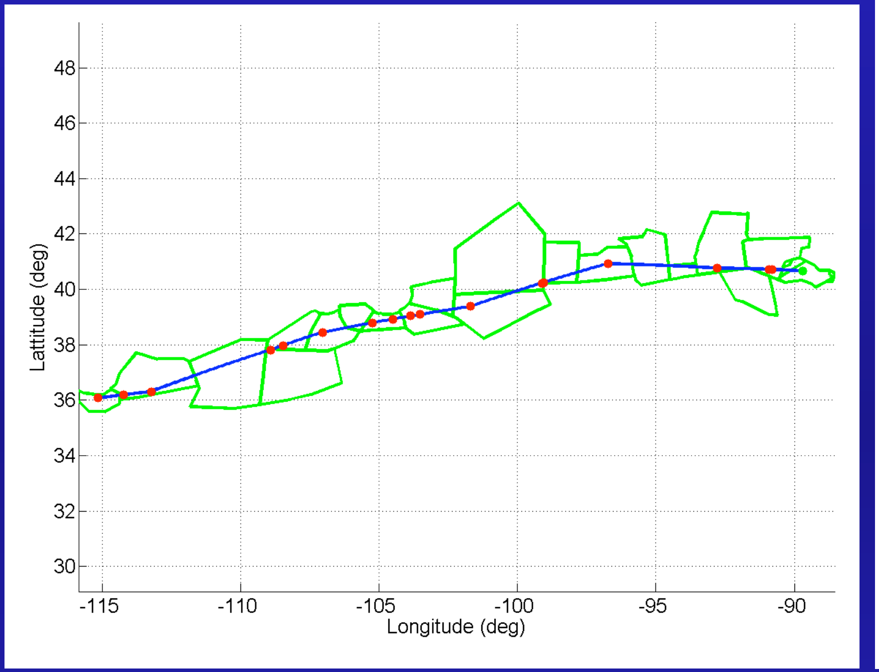


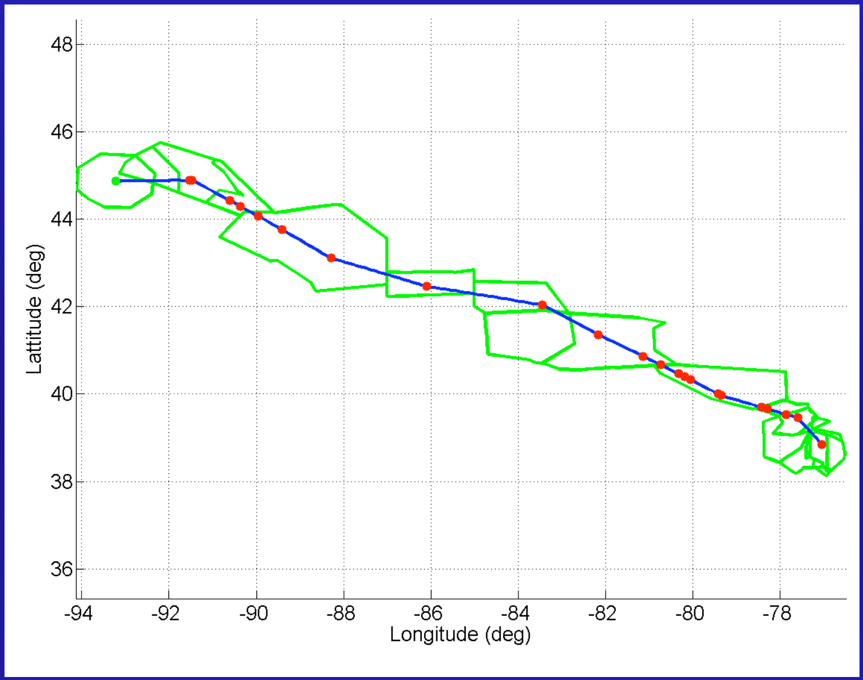
Check if Aircraft in Sector

- Check Within Bounding Box
- Ray Casting Algorithm

- 3D Hash Map: 2° Lat, 2° Lon, 1000 ft.
- For every cell, store all sectors whose bounding boxes overlap the cell

Results of Sector Identification Code





Sequence of Sectors
Traversed by Flight NWA1715

Sequence of Sectors
Traversed by Flight AAY462

Modeling Traffic Flow

- Aggregate models
  - Flights aggregated into flows
  - Optimal controls determine rates for entering/exiting flights
  - Solution needs disaggregation for implementation in reality
  - In general, computationally efficient
- Aircraft-level models
  - Controls considered for each flight
  - Solution is “implementation ready”
  - In general, computationally difficult
- Both approaches rely on airspace simulations