Cal Poly, **Computer Science**
clupo@calpoly.edu

High Performance Ocean Modeling using CUDA

Chris Lupo
Computer Science
Cal Poly

Cal Poly, **Computer Science**
clupo@calpoly.edu

Acknowledgements

- Dr. Paul Choboter
- Jason Mak
- Ian Panzer
- Spencer Lines
- Sagiv Sheelo
- Jake Gardner

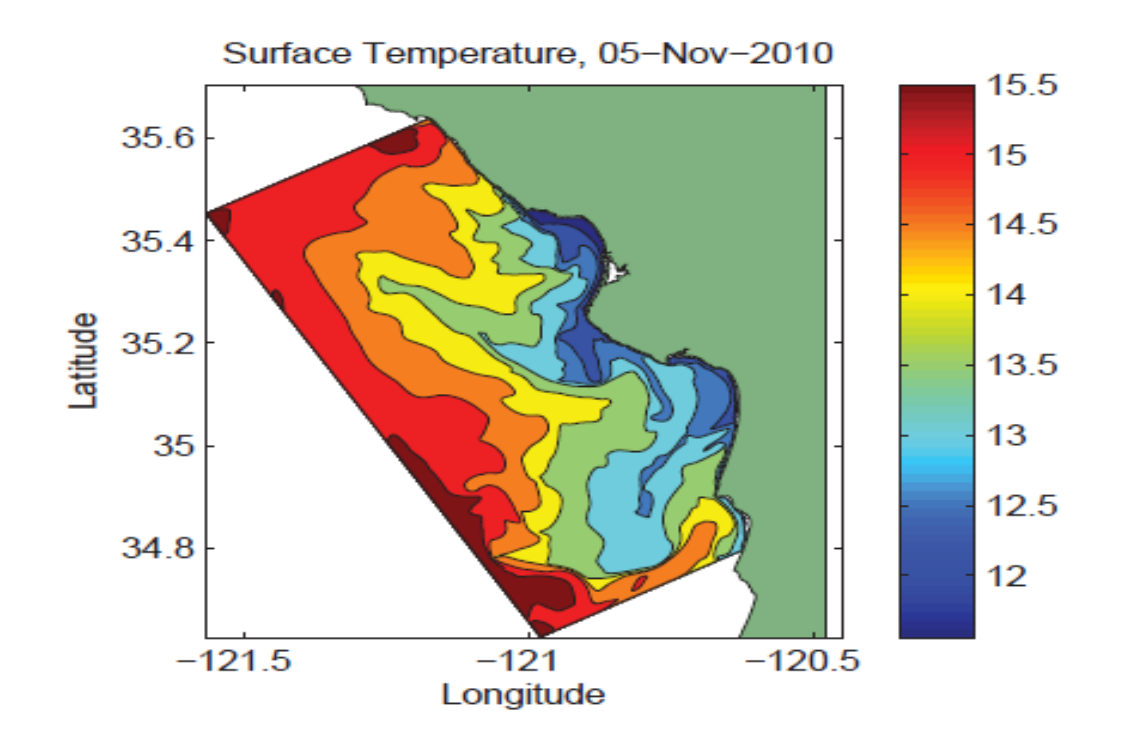
Cal Poly, **Computer Science**
clupo@calpoly.edu

Background

- Joint research with Dr. Paul Choboter (Cal Poly, Math)
  - He's the Oceanographer
  - Research is focused on dynamics of wind-driven coastal upwelling
  - Model cross-shore and alongshore transport in the California coastal ocean
  - Useful for studying applications such as:
    - pollution transport
    - algal blooms
    - sediment transport

Cal Poly, **Computer Science**
clupo@calpoly.edu

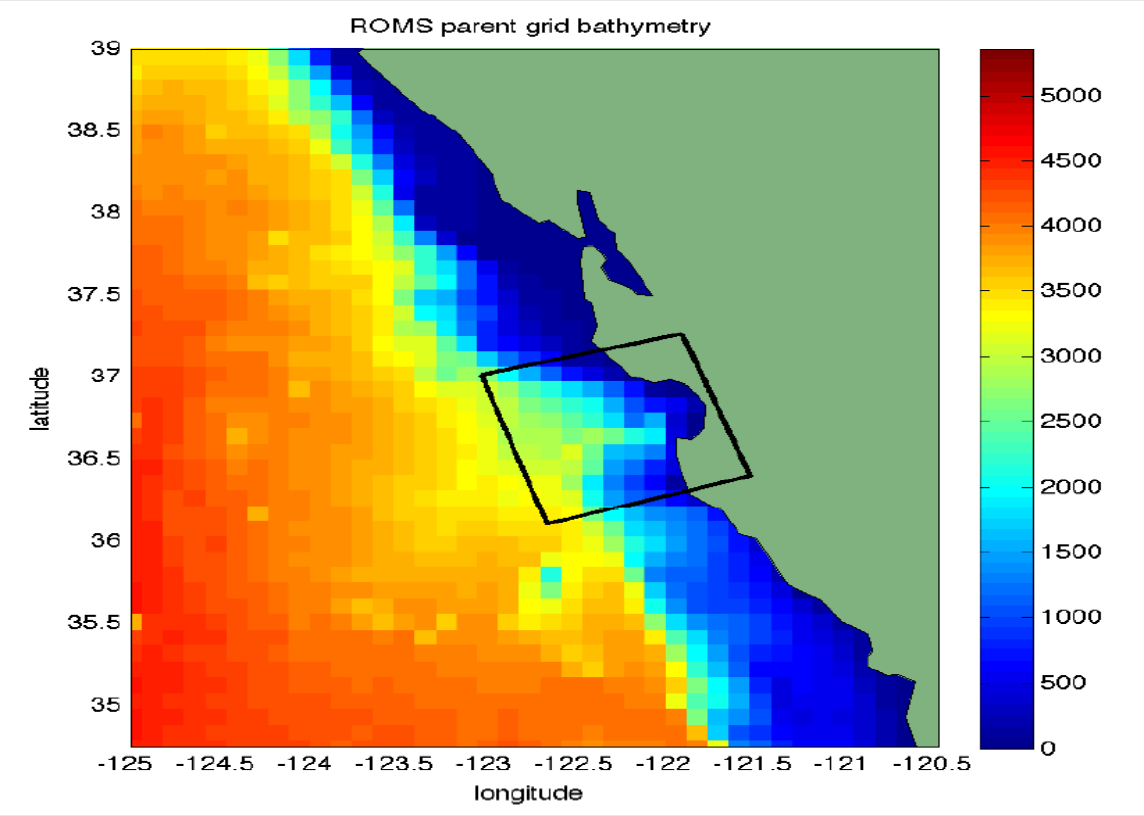
Background



- Surface temperature modeling, San Luis Obispo County

Cal Poly, **Computer Science**
clupo@calpoly.edu

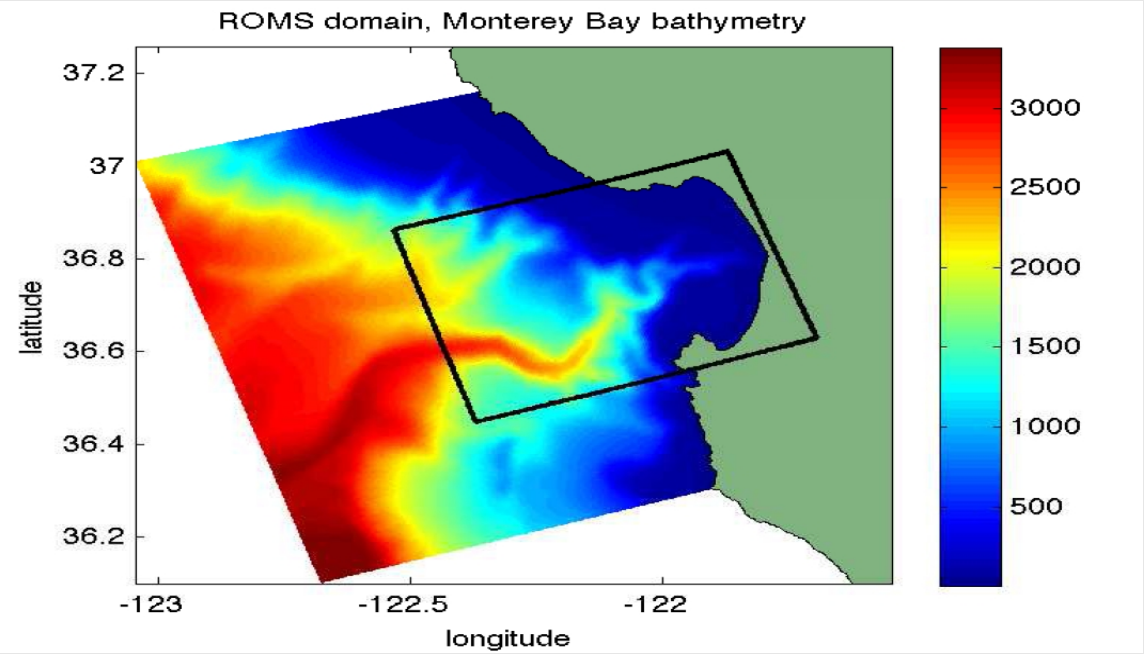
Background



- Coarse-grained bathymetry modeling of San Francisco and Monterey Bays
- ~10 km grid resolution

Cal Poly, **Computer Science**
clupo@calpoly.edu

Background



- Finer-grained bathymetry modeling of Monterey Bay
- ~1 km grid resolution

Cal Poly, **Computer Science**
clupo@calpoly.edu

Objectives

We want more resolution!

Ideally we would have meter or sub-meter grids

- Creek mouths
- Geologic / topologic features (e.g. Monterey Canyon)

Existing models can take days or weeks to compute

Cal Poly, **Computer Science**
clupo@calpoly.edu

ROMS

Regional Ocean Modeling System (ROMS)

Robust, actively developed open-source numerical ocean modeling software used by the scientific community

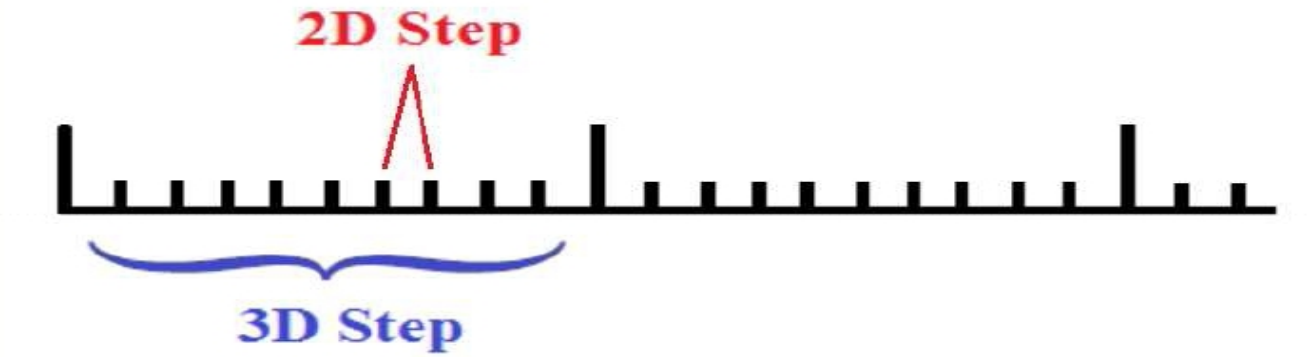
Developed in Fortran 90/95, ~400k loc

Staggered, finite, difference grid:

- velocity, temperature, salinity, and sea surface elevation

Cal Poly, **Computer Science**
clupo@calpoly.edu

ROMS



Iterative time-stepping algorithm

Simulation time interval: large (slow), depth-dependent time step (baroclinic) solves 3D equations, uses many short (fast), depth integrated time steps (barotropic) to solve 2D equations

Short time steps (step2D) occupy more than 50% of runtime

Cal Poly, **Computer Science**
clupo@calpoly.edu

Parallelism

ROMS has been developed with two parallel computing paradigms

- Distributed with message passing for clusters
- Shared memory with OpenMP for multicore

Cal Poly, **Computer Science**
clupo@calpoly.edu

Parallelism: Distributed

MPI

Great for multi-node clusters

Advantages:

- Very scalable, supports large number of processors
- Fine-grained parallelism

Disadvantages:

- Cost: Can easily hit $100k for entry-level system
- Maintenance: need a machine room
- Support: third-party contracts, technicians

Cal Poly, **Computer Science**
clupo@calpoly.edu

Parallelism: Shared

OpenMP

Great for desktop researchers

Advantages:

- Cost: Great performance for < $10k
- Maintenance: possible for end-user
- Support: possible for end-user

Disadvantages:

- Not very scalable
- Course-grained parallelism

Cal Poly, **Computer Science**
clupo@calpoly.edu

Parallelism: Shared

OpenMP

Great for desktop researchers

Advantages:

- Cost: Great performance for < $10k
- Maintenance: possible for end-user
- Support: possible for end-user

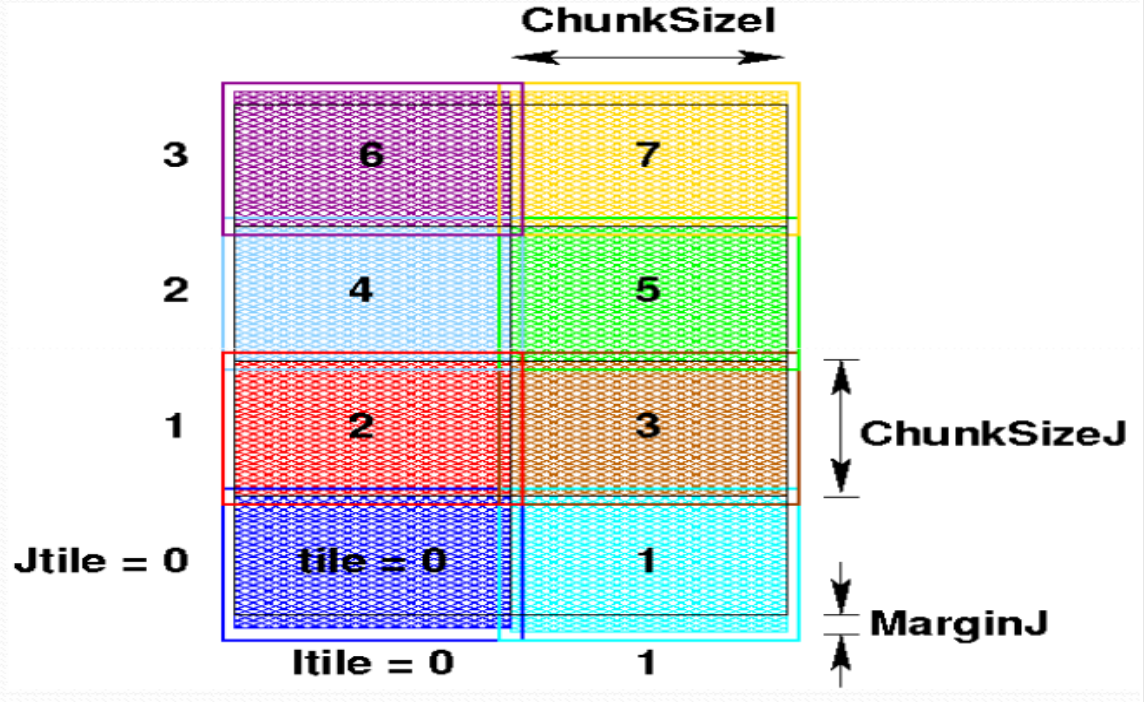
Disadvantages:

- Not very scalable
- Course-grained parallelism

Cal Poly, **Computer Science**
clupo@calpoly.edu

OpenMP Implementation

Partition the grid into tiles and assign each tile to a core (thread)



http://www.myroms.org/

Cal Poly, **Computer Science**
clupo@calpoly.edu

CUDA Approach

Re-use the existing OpenMP tiling

Partition grid into small tiles, assign each tile to thread on GPU

Use the smallest possible tile, 2x2, to saturate GPU

Cal Poly, **Computer Science**
clupo@calpoly.edu

CUDA Approach

Rewrite the step2D function as a CUDA kernel

Allocate GPU memory once

Transfer memory before and after all step2D computations

Cal Poly, **Computer Science**
clupo@calpoly.edu

CUDA Approach

```
DO my_iif=1,nfast(ng)+1
[...]
!$OMP PARALLEL DO
  DO thread=0,numthreads-1
    subs=numtiles/numthreads
    DO tile=subs*thread,subs*(thread+1)-1,+1
      CALL step2d (ng, tile)
    END DO
  END DO
!$OMP END PARALLEL DO
[...]
END DO
```

(a) OpenMP

```
CALL step2d_host_to_device()
DO my_iif=1,nfast(ng)+1
[...]
  CALL step2d_kernel<<<dim_grid, dim_block>>>
   (num_tiles,krhs(ng),kstp(ng),knew(ng),
    nstp(ng), nnew(ng),PREDICTOR_2D_STEP(ng),
    iif(ng), Lm(ng), Mm(ng), iic(ng),
    nfast(ng),dtfast(ng), ntfirst(ng),
    gamma2(ng), rho0, work_dev)
[...]
END DO
CALL step2d_device_to_host()
```

(b) CUDA

Cal Poly, **Computer Science**
clupo@calpoly.edu

Porting Challenges

step2D function > 2000 lines

Lack of encapsulation in Fortran code

Necessary global variables must all be identified

Index translation

Cal Poly, **Computer Science**
clupo@calpoly.edu

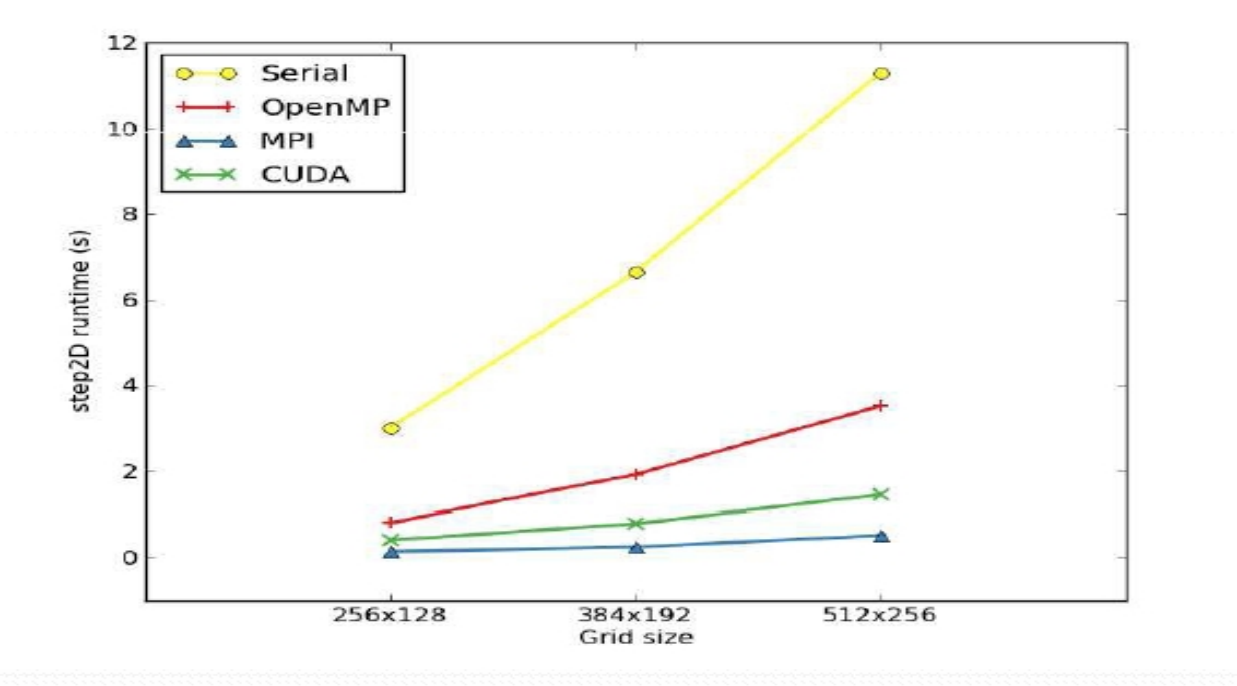
Experimental Setup

- OpenMP
  - 2 Intel Xeon E5504 (8 cores total)
- MPI
  - Intel Xeon 5130 processors (64 cores total)
- CUDA
  - Tesla C2050 (448 cores)

Cal Poly, **Computer Science**
clupo@calpoly.edu

Results

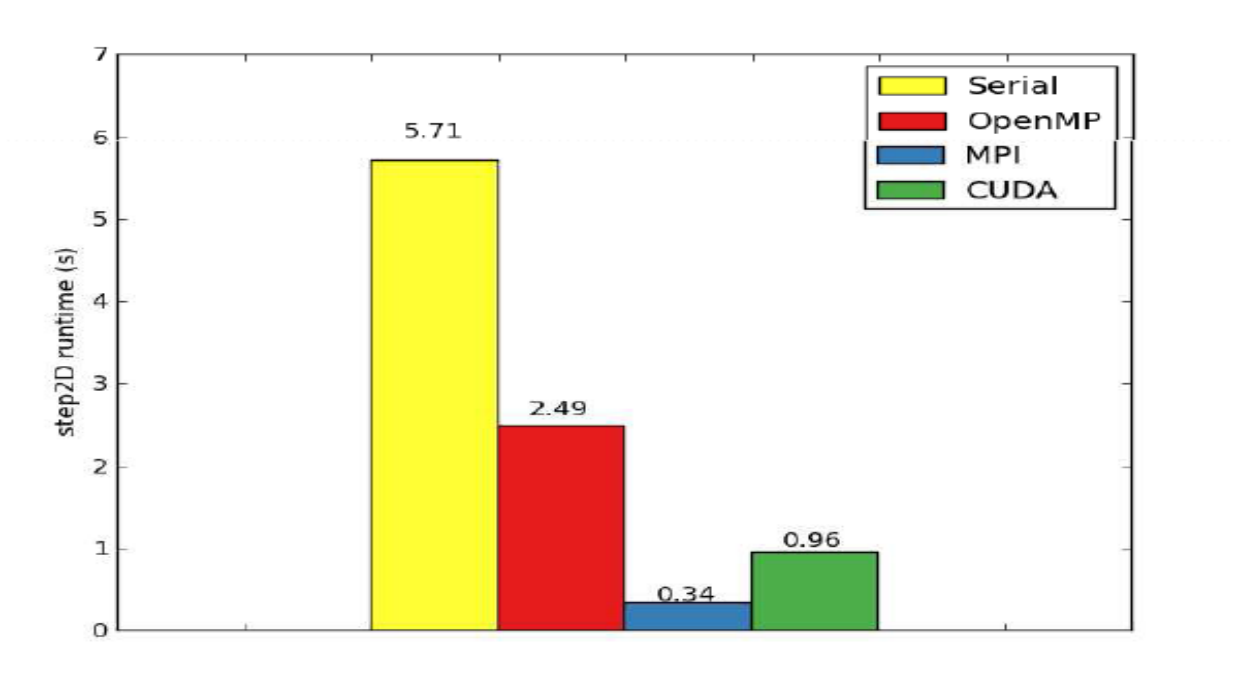
Upwelling simulation: internally specified idealized conditions



Cal Poly, **Computer Science**
clupo@calpoly.edu

Results

Observational data from CA central coast (512x256)



Cal Poly, **Computer Science**
clupo@calpoly.edu

Next Steps

- Run more portions of the simulation on the GPU, this is only step2D
  - OpenACC looks promising, but we've struggled with it
- GPU optimizations: faster memory, double-buffering, divergence removal
- Multi-GPU support
- Hybrid OpenMP+CUDA Implementation
- Utilize newer hardware to exploit dynamic parallelism

Cal Poly, **Computer Science**
clupo@calpoly.edu

Conclusions

GPUs show promising results for accelerating ocean modeling software

Open new doors in simulation size or time scale for research. Can allow us to do better desktop science.

Cal Poly, **Computer Science**
clupo@calpoly.edu

Conclusions

GPUs show promising results for accelerating ocean modeling software

Open new doors in simulation size or time scale for research. Can allow us to do better desktop science.

Cal Poly, **Computer Science**
clupo@calpoly.edu

Questions?