High Performance Computing Systems (CMSC714)

Lecture 23: Parallel Discrete-event Simulation

Abhinav Bhatele, Department of Computer Science

UNIVERSITY OF MARYLAND

Announcements

- Project demos: December 3 and 5
- Final project due on: December 11, 5:00 pm

Summary of last lecture

- *n*-body problem: gravitational forces on celestial bodies
- Several parallel algorithms:
  - Barnes-Hut
  - Fast Multiple Method
  - Particle Mesh
  - P3M
- Simulation codes: FLASH, Cello, ChaNGa, PKDGRAV

Discrete-event simulation

- Modeling a system in terms of events that happen at discrete points in time
- Either model discrete sequence of events
- Or model time-stepped sequences
- Simulation typically involves system state, event list and a global time variable

Parallel discrete-event simulation

- Divide the events to be simulated among processes
- Send messages wherever there are causality relationships between events
- Synchronize global clock periodically

Conservative vs. optimistic simulation

- Conservaties DES
  - Do not allow any causality errors
- Optimistic DES
  - Allow causality errors and rollback if needed

Epidemiology simulations

- Agent-based modeling to simulate epidemic diffusion
- Models agents (people) and interactions between them
- People interact when they visit the same location at the same time
- These "interactions" between pairs of people are represented as "visits" to locations
- A bi-partite graph of people and locations is used

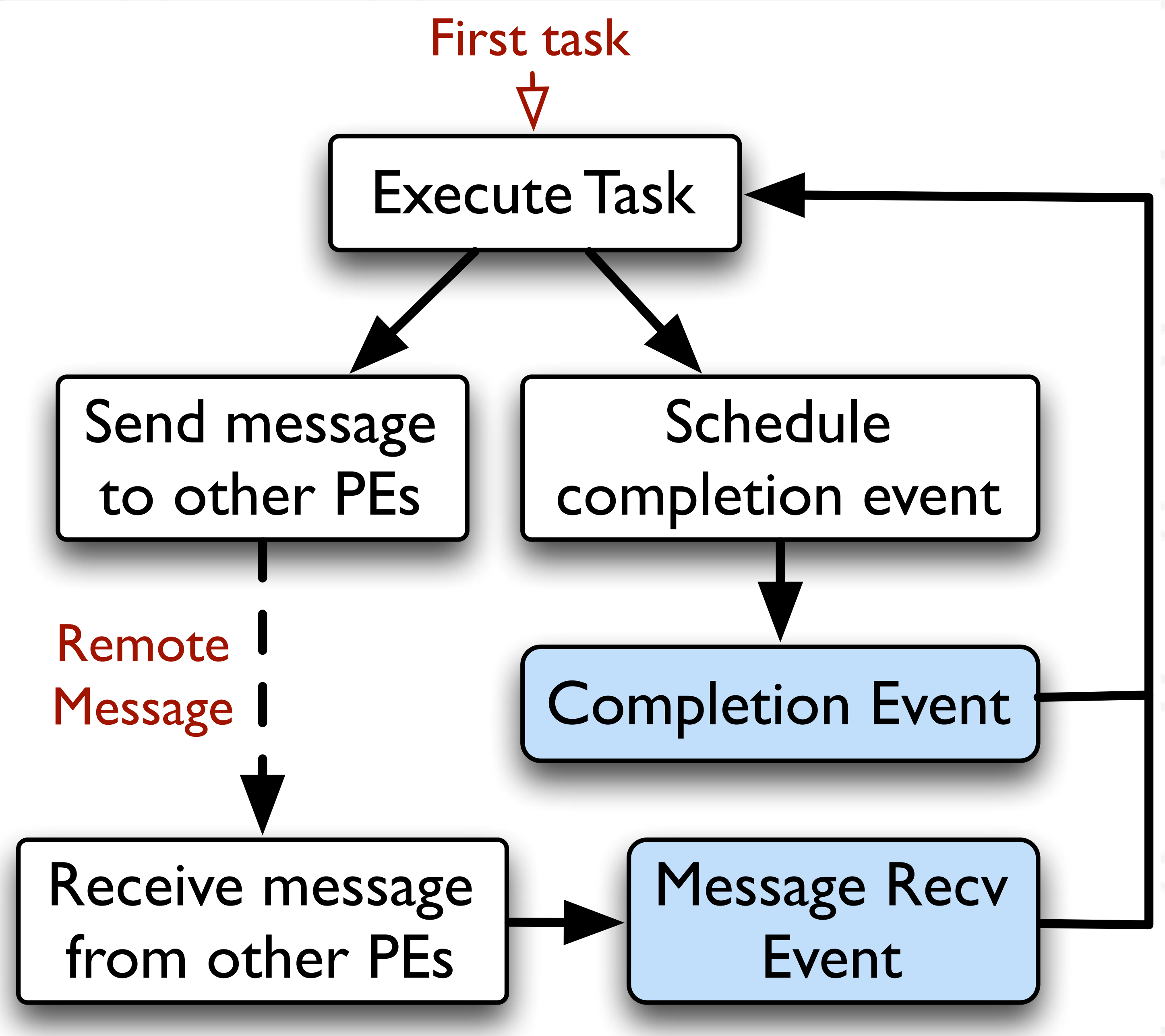
EpiSimdemics: Parallel implementation

- All the people and locations are distributed among all processes
- Computation can be done locally in parallel
- Communication when sending visit and infection messages
- Uses Charm++, a message-driven model

```
while d ≤ d_max do
    for p ∈ P do
        Evaluate scenario trigger conditions;
        Update health state h_p, if necessary, and reevaluate triggers;
        foreach v ∈ V_p ( visit schedule of p) do
            Send visit message m to location l;
        end
    end
    for l ∈ L do
        foreach m destined for l do
            Determine the sublocation l_s to visit;
            Create an arrival and departure event for each visit;
            Put the events into the event queue q_e of l;
        end
        Reorder q_e by the time of event in ascending order;
        foreach e ∈ q_e do
            if e is arrival then
                Put p into sublocation l_s;
            else
                Remove p from sublocation l_s;
                foreach p' currently in l_s do
                    Compute disease transmission probability q
                      between p' and p;
                    if q > threshold then
                        Send infection message to the infected
                          person (p or p');
                    end
                end
            end
        end
    end
    d++;
end
```

Trace-driven network simulation

- Task is started at time t_s
- Completion event scheduled for time $t_s + t_e$
- Possible remote messages to other PEs
  - Kick off other tasks that depend on a message



Running TraceR in optimistic mode

- Record extra information during forward execution to enable rollback later
  - List of tasks triggered by a message recv or completion event
- Implement reverse handlers for each event

Questions

Preliminary Evaluation of a Parallel Trace Replay Tool for HPC Network Simulations

- Is there a reason why rollback efficiency is calculated as a negative score?
- In the intro, the paper describes one of the weaknesses of current DES-based network simulators as only simulating "synthetic communication patterns". What exactly is meant by this?
- Is the optimistic mode a unique concept to TraceR? Or is it commonly implemented in tools that execute on instruction traces?

Questions

Overcoming the Scalability Challenges of Epidemic Simulations on Blue Waters

- It says receivers have no prior knowledge of expected messages and this turns process into a slower BSP, but locations do have access to the people they are connected to. Is it more expensive to send a message like "I'm not visiting today" per person to each connected location? so then locations can check all messages to see whose messages is not send yet.
- We usually say charm is suited for over decomposed problems, but is there a minimum limit for this over decomposition? because paper mentions an overhead.
- What is a sublocation? I don't quite understand how exclusive sets interact with each other in the same location? like 4th and 5th nodes in Figure 6.
- For the Charm SMP mode section: I don't quite follow how this creates more communication threads/cores? say n is 12 and k is 4, does it mean there are 4 communication/OS processes and 4 compute threads?
- Do the government agencies develop these models like hierarchical social network? or CS people develop them then government chooses one of them?
- How these simulations are used? do they stop the simulation make an intervention at some point and then fork the simulation to see the effect of it? or are they just used to get a sense of how dangerous a disease with a new transmission function?
- How do we validate these simulations or transmission functions?
- The paper describes METIS as a tool that "allows users to specify the load balance constraint in terms of the tolerance variable in the sum of vertex weights per partition". Exactly how does this work?
- Can you talk a little bit about how the completion detection mechanism works? The text says that "completion is detected when the participating objects have produced and consumed an equal number of messages globally" yet I had been under the impression that this communication of messages may be non-deterministic.
- What are some of the benefits and downsides to the two buffer flushing mechanisms (per-buffer flushing vs space-wise flushing)

Questions?

UNIVERSITY OF MARYLAND

Abhinav Bhatele

5218 Brendan Iribe Center (IRB) / College Park, MD 20742

phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu