Recursive Propagation Scheduling for Holonic Systems

Scott Logie

Simon Fraser University
School of Engineering Science
Burnaby, BC Canada

February 27, 2003

Research supported by **NSERC**, **ASI**

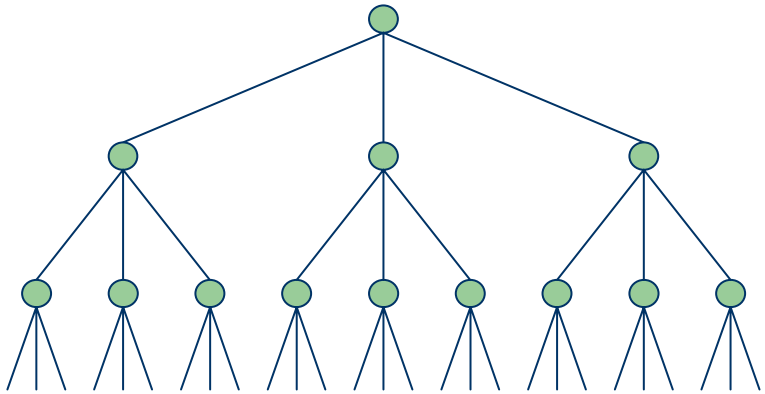
Introduction

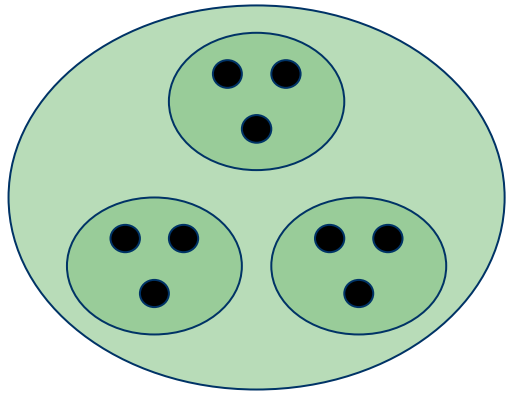
- Motivation
- Research Goals
- Problem Definitions
- Algorithm Details
- Simulations Results
- Further Development
- Questions

Holonic Manufacturing System

- Holon Definition
  - Arthur Koestler (1967) – *hol* whole; –*on* particle
- Production facilities with multiple participants
  - Distribute system knowledge among constituents
  - Centralized vs. Decentralized
- Holon vs. Multi-Agent Systems
  - Holon goals – obtain feasible schedule
  - System goals – minimize time to completion

Holonic Systems





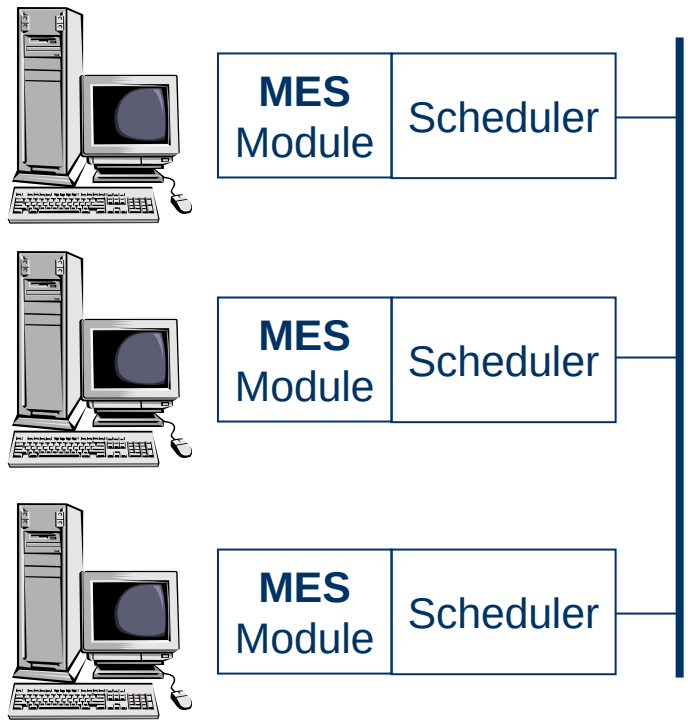
- HMS Consortium Goals
  - Stability in the face of disturbance
  - Adaptability in the face of change
  - Efficient use of available resources

Research Goals

- Implement the Recursive Propagation Scheduling Technique [Hino98] in C++
- Compare performance with Hino's results and improve
- Implement in Java
- Interface with a manufacturing execution system (MES)

Manufacturing Execution Systems

- Keeps track of production schedules, inventory availability, works in progress
- Provides a real-time view of plant operations

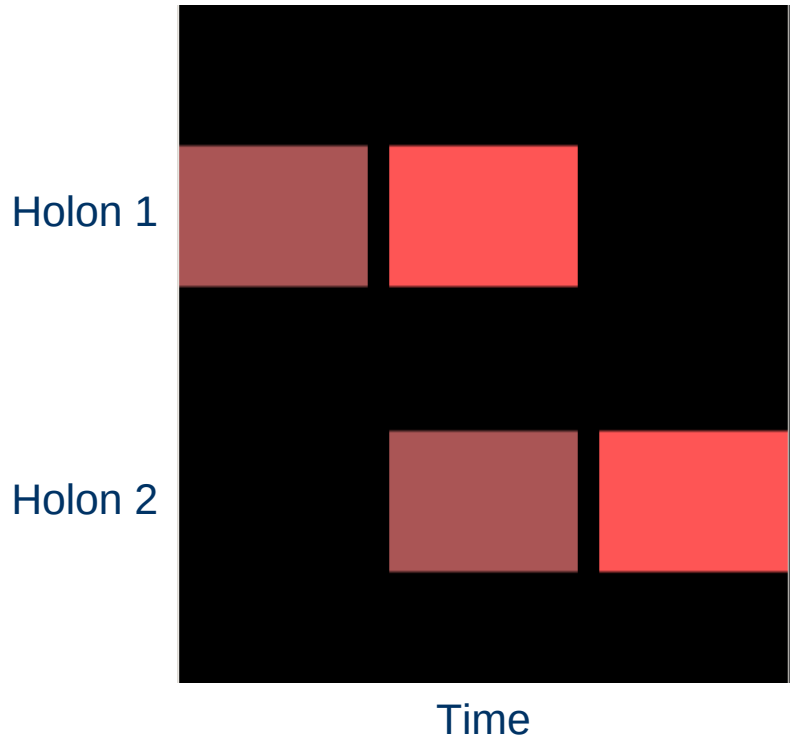


Problem Definitions

- Process
  - A task that is performed by one holon
- Operation
  - Consists of a series of processes
  - Distributed among holons – Never revisits holon

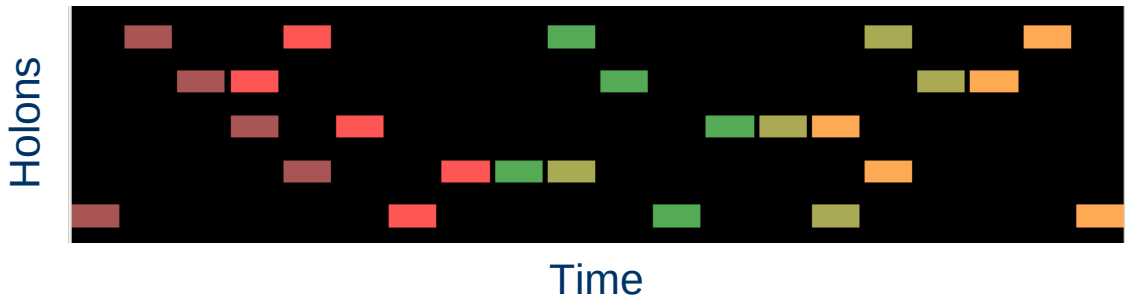
Recursive Scheduling Technique

- Decentralized algorithm
- Select target operation
- Notification of change
  - Recursively passed to subsequent holons
- Report results
  - Inform originator of impact of all changes

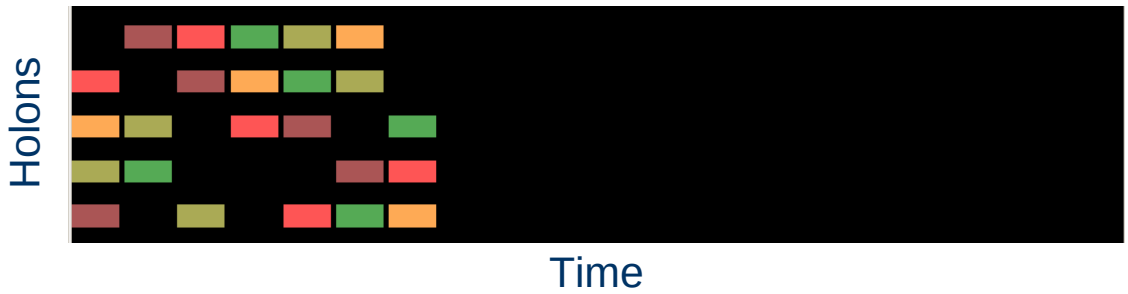


Algorithm Details

- Finding a feasible solution

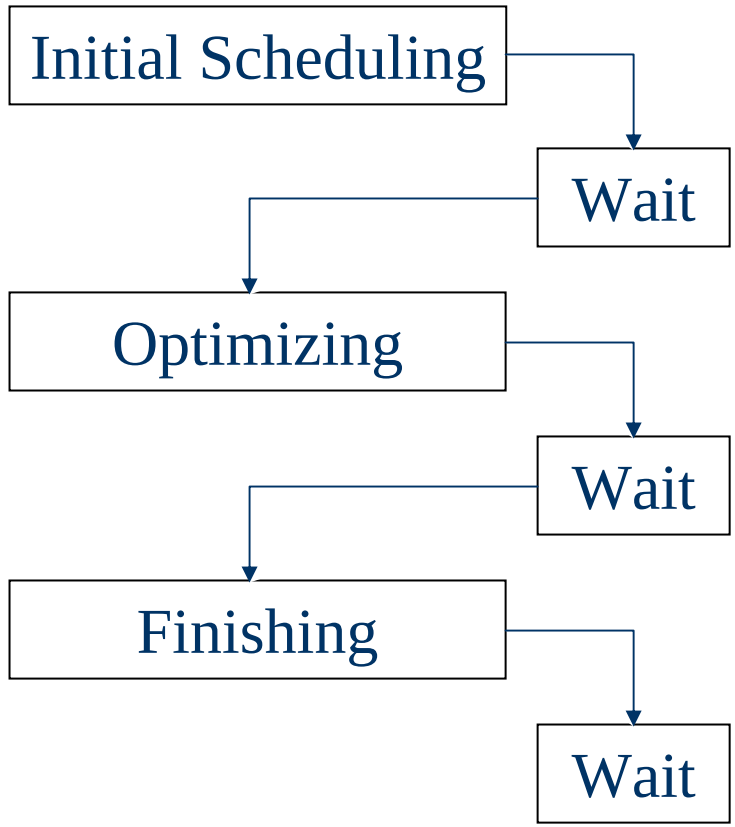


- Optimize the solution by reordering processes



Holonic State Machine

- Agree on feasible schedule
- Wait for others
- Optimize schedule until satisfied
- Wait for others
- Finish last operation

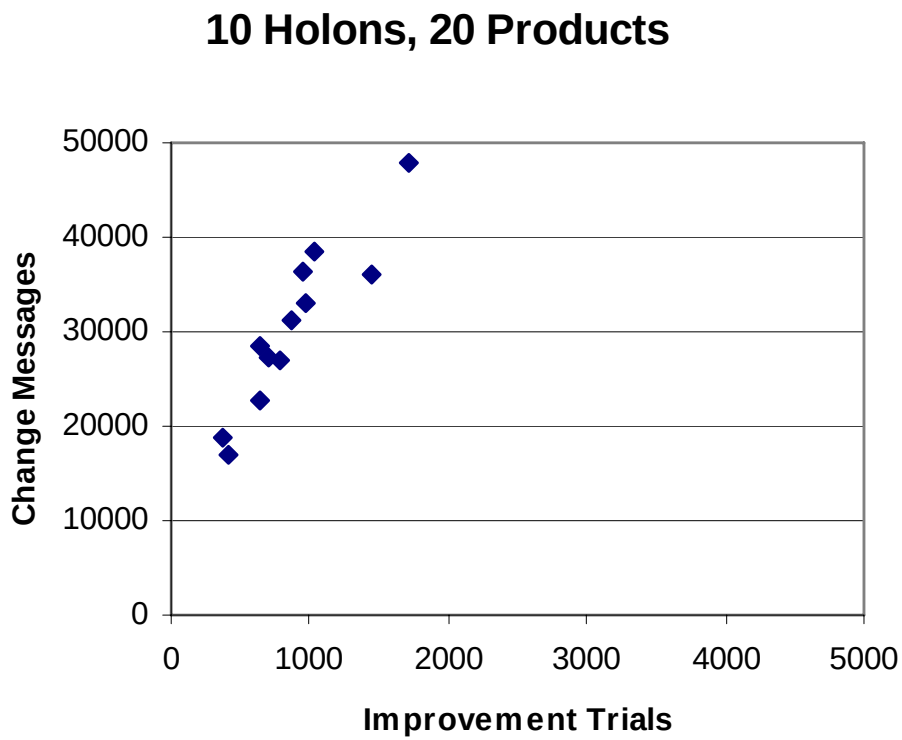


Improvements on Hino's Technique

- No details about optimization method – reordering of processes
- Smarter swapping
  - Large changes first
- Smarter stopping condition
  - When all processes have been addressed with no change to the current holon's schedule

Simulation Results

- Multi-threaded C++ app with OpenGL display
- Hino's benchmark: 10 holons schedule 20 products in 5000 trials
- My simulations: 20 products for 10 holons in fewer than 900 trials on average



Further Developments

- Port scheduler code from C++ to Java
- Allow re-scheduling after agent failure
- Verify simulations over networked workstations
  - JADE - **J**ava **A**gent **DE**velopment Framework
  - Multi-agent environment implemented in Java
- Attach scheduler module to existing manufacturing execution system

Questions

Please contact me with any additional inquiries

Scott Logie, MASc Student
Simon Fraser University
c/o School of Engineering Science
Burnaby, BC Canada
slogie@sfu.ca