Adaptive Communication Scheduling

Greg Willden, Jeremy Price, Denise Varner, Ben Abbott
-- Southwest Research Institute (SwRI)

Djaffer Idaroudene
-- St. Mary’s University

Work partially funded by DARPA MoBIES contract # F33615-01-C-1909
Outline

• Objective

• Background
  – Real-time Scheduling
  – ASC scheduler
  – Model Integrated Computing
  – CPU scheduling model -- MoBIES inspired – Avionics

• Communication Scheduling
  – Signal Exploitation inspired

• Conclusion
Objective

- Provide a composable framework for real-time scheduler synthesis

  - Support Runtime Adaptation based on:
    - *Integration* with scheduler/system objectives
    - *Modeling* of performance tradeoffs
    - *Combining* CPU and communication scheduling

  - Transition smoothly to/from classic scheduling paradigms
    - Provable
    - Extensible
Generic Real-Time Scheduler

Scheduling Policy
- Static – RMS
- Dynamic – EDF
- etc.

Scheduler State
Task 1 State
Task N State

Selector
choose highest
context-switch

Perform Task

– Allocation of time and resources to tasks such that timing constraints are met
– Timing constraints range from Soft to Hard
• Always schedules the (ready) task with the highest arrival rate
• Priorities are static

• Graph of task priority

• Ignores additional system parameters
• Starves lowest rate task in overload (often regardless of importance)
Scheduling Policy
Earliest Deadline First (EDF)

- Always schedules the (ready) task with the earliest deadline
- Priorities are dynamic
- Several similar dynamic approaches -- example least slack

- Graph of task priority

- Ignores additional system parameters
- Misses all task deadlines in overload (regardless of importance)
- Various policies for missed deadlines might help, but which one?
• **Want scheduling policy that incorporates system parameters**

  – *Adaptive Service Coordination (ASC) Scheduler*
    - Describe scheduler coordination using generic information in the form of:
      - equations, state machines, and rules
      - include application specific *system parameters*

    • Synthesize domain specific real-time schedulers for distributed systems from models

    • Support provable and heuristic-based schedulers

    • Utilize a model-based approach for scheduling policy integration
ASC Scheduler

Multi-Aspect System Model

Specialized Distributed Scheduler

Timing requirements aspect

Futile
Tight
Loose

slack

0.0

1.0

Scheduler State

Task 1 State

Task n State

Filter

FAM

Selector

Task to Perform

Salient State

Priority
Best Effort Scheduler -- Model

- GME model used to seed Matlab fuzzy toolbox interface
- Matlab additions to the model verified and integrated within the overall GME model
Best Effort Scheduler Model

Rules & Priority Surface

1: tacticalDisplay: schedule==marginal) \& (navDisplay: schedule==ontime) => (navDisplay: priority: delta=lower)

ASC Scheduler Characteristics

- Implements classic schedulers
  - RMS, LS, EDF, PDP, TTP, distributed GRMS,…

- Transitions from deterministic to heuristic or stochastic solutions

- Provides interface for automatic program generator specialization
Domain-Specific Modeling

Multiple-Aspect Domain/Target-Specific Generators (including ASC scheduler in two places)

SW Execution Environment

Analysis and Verification Tools
ASC-ESML (modeling)

- Enables modeling of scheduling policy state machine:
  - State definitions
  - Scheduling policies for each state (per component)
  - State transitions
  - Templates supplied for standard schedulers (RMS, EDF,...)

- Integrated with Matlab fuzzy logic toolbox as GUI editor for developing specific scheduling policies

- Input: ESML model, ASC model
- Output: Scheduler Parameters
- Metamodel defines paradigm
• **ASC-Scheduler (runtime)**
  – Utilizes ASC-ESML synthesized priority calculation model
    • Supports dynamic and adaptive scheduling policies
    • Coordinates state transitions based on modeled triggers

  – Integrated with ACE/TAO & Boeing Bold Stroke
    • Supports preemptive scheduling of component execution
    • Support scheduling of network communication

  – Instrumented
    • Boeing format and VxWorks Windview format

  – **Input:**
    • Scheduler generated by ASC-ESML

  – **Outputs:**
    • Scheduled system
    • Instrumentation
Communication Scheduling
Signal Exploitation (Interaction Diagram)

Processor 2
Signal Exploitation

(communication scheduler connecting a schedule)
Signal Exploitation (communication scheduler inputs)
Signal Exploitation

(communication scheduler rules and output)

1. If (msgSize is small) and (qLength is optimum) then (Priority is high) (1)
2. If (msgSize is small) and (qLength is empty) then (Priority is high) (1)
3. If (msgSize is small) and (qLength is full) then (Priority is low) (1)
4. If (msgSize is large) and (qLength is full) then (Priority is high) (1)
Signal Exploitation
(communication scheduler resulting control surface)

- **normal queue**: smaller messages [alarm display] correspond to higher priorities.
- **queue large**: larger messages [spectrum display] higher priorities
  (to help drain the queue faster)
Summary

• Adaptive Scheduling
  – Allows system parameters to be considered
  – Communication/CPU Scheduling is integrated
  – Fuzzy reasoning supports imprecise global state scheduling

• Performance
  – Slight additional cost for the ASC
  – Adjusted for domain specific scenarios

• Timing guarantees
  – Traditional techniques are still valid
  – Provides possibility for synthesis or runtime to choose technique

• Complexity of design time modeling
  – Increased, but allows more precision
• Integrated with ASC CPU scheduler

• Implements classic schedulers
  – TTP, PDP, distributed GRMS (proofs available)

• Allows for use and transmission of imprecise global state through the fuzzy reasoning
**Details**

- LV = linguistic variables same set available for both CPU and Comm scheduler
- Core scheduler priority evaluation code same as CPU scheduler
Four new components added to GME:

- **ASC**: Interpreter to generate scheduler parameters based on the scheduler modeled.
- **BUILD**: Compiles and links the specialized ASC scheduler with the modeled OEP scenario.
- **FIS**: Utilizes matlab fuzzy toolbox as an integrated editor for the ASC scheduler model.
- **RUN**: Executes the OEP scenario, collects timing (performance statistics) and displays summary information.
ASC Modeling Aspect in GME

- Integrated with GME releases
- Functional Interpreter
- Integrated with Matlab for fuzzy portion of the model
ASC Execution Results

Timing results of the instrumented run can be shown in a variety of ways:

- WindRiver’s Tornado/VxWorks tool Windview timelines
- Textual summary files
- Other tools possible based on the instrumentation interface metamodel