SPRINT: Extending RUN to Schedule Sporadic Tasks

Andrea Baldovin, Geoffrey Nelissen, Tullio Vardanega, Eduardo Tovar

22nd International Conference on Real-Time Networks and Systems
Versailles, October 10th, 2014
Outline

1. Recalling RUN
2. Introducing SPRINT
   - The intuition behind
   - Budget computation
   - Server priorities
   - Example
3. Simulation results
4. Future work
Optimal, semi-partitioned scheduling algorithm for multicore

Consists of 2 phases

- **Offline**
  * Builds a reduction tree by dual scheduling and bin-packing

  

  ![Dual Schedule Diagram]

  ![Primal Schedule Diagram]

  $U(T^*) = n - U(T)$

- **Online**
  * Solves each single-core scheduling problem with an optimal algorithm (EDF)

Scheduling $m$ tasks on $n$ processors
  $\rightarrow k$ scheduling problems on 1 processor
RUN - Example (Offline phase)

\[ T \]

- \( \tau_1 \) with \( U(\tau_1) = 0.6 \) and \( <3.6,6> \)
- \( \tau_2 \) with \( U(\tau_2) = 0.6 \) and \( <3.6,6> \)
- \( \tau_3 \) with \( U(\tau_3) = 0.3 \) and \( <3,10> \)
- \( \tau_4 \) with \( U(\tau_4) = 0.3 \) and \( <4.5,15> \)
- \( \tau_5 \) with \( U(\tau_5) = 0.6 \) and \( <1.8,3> \)
- \( \tau_6 \) with \( U(\tau_6) = 0.3 \) and \( <4.5,15> \)
- \( \tau_7 \) with \( U(\tau_7) = 0.3 \) and \( <9,30> \)
RUN - Example (Offline phase)

\[ \begin{align*}
S^0 & \quad \text{U = 0.6} \\
S^0_1 & \quad \tau_1 \quad \text{U(\tau_1) = 0.6} \quad <3.6,6> \\
S^0_2 & \quad \tau_2 \quad \text{U(\tau_2) = 0.6} \quad <3.6,6> \\
S^0_3 & \quad \tau_3 \quad \text{U(\tau_3) = 0.3} \quad <3.10> \\
S^0_4 & \quad \tau_4 \quad \text{U(\tau_4) = 0.3} \quad <4.5,15> \\
S^0_5 & \quad \tau_5 \quad \text{U(\tau_5) = 0.6} \quad <1.8,3> \\
S^0_6 & \quad \tau_6 \quad \text{U(\tau_6) = 0.3} \quad <4.5,15> \\
S^0_7 & \quad \tau_7 \quad \text{U(\tau_7) = 0.3} \quad <9,30>
\end{align*} \]
RUN - Example (Offline phase)
RUN - Example (Offline phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)
RUN - Example (Online phase)

Graphical representation of the online phase with states and transitions labeled with probabilities and time intervals.

RUN only fits for *periodic* task sets
- Job release events at level 0 only occur at multiples of tasks’ activations
- However, artificial workload may be present in the system to reach full utilization assumed by the packing process
- Let’s take advantage of this time for trying to accommodate sporadic activations instead
- In this scenario, job releases may occur at any point in time
- Recompute server execution budgets accordingly to account for future sporadic releases
At level 0, adjusting budgets of servers is all we need.
SPRINT - Reduction at level 0

\[ \text{bdgt}(S^0) \]

\[ \text{bdgt}(S^{0*}) \]

\[ \tau_3 \]

\[ d(S^0_3) \]
SPRINT - Reduction at level 0

\[ \text{bdgt}(S_0^0), 0 = (d(S_3^0) - 0) - \sum_{\tau_i \in S_3^0} U(\tau_i) \times (d(S_3^0) - 0) \]

\[ \text{bdgt}(S_3^0, 0) = \sum_{\tau_i \in S_3^0 \cap A(0)} U(\tau_i) \times (d(S_3^0) - 0) \]

\[ \text{bdgt}(S_0^0) \]

\[ \text{bdgt}(S_0^0) \]
SPRINT - Reduction at level 0

\[ \text{bdgt}(S^0) \]

\[ \text{bdgt}(S^{0*}) \]

\[ \tau_3 \]

\[ \tau_4 \]

\[ d(S^0_3) \]
SPRINT - Reduction at level 0

\begin{align*}
\text{bdgt}(S_0^0), t) &= (d(S_3^0) - t) - \text{bdgt}(S_3^0, t) - \sum_{\tau_j \in S_3^0 \setminus A(t) \cap \text{Rel}(t)} U(\tau_j) \times (d(S_3^0) - t) \\
\text{bdgt}(S_0^0) &= \text{bdgt}(S_3^0, t) + \sum_{\tau_j \in S_3^0 \cap \text{Rel}(t)} U(\tau_j) \times (d(S_3^0) - t) \\
\text{bdgt}(S_0^0) &= \tau_3 \quad \tau_4 \quad \tau_5 \quad \tau_6 \quad \tau_7
\end{align*}
Server priorities

- But that’s not enough when the full tree is considered
We need a mechanism to select the "right" branch of the tree.
**Observation 1:** $S_{i}^{1*}$ executes $\iff$ all $S_{k}^{0} \in S_{i}^{1}$ execute
Observation 2: \( S_i^{1*} \) does not execute \( \iff \) all \( S_k^0 \in S_i^1 \) but one execute.
Server priorities

- **Inference**: Therefore, we don’t need to execute $S_i^{1*}$ if some sporadic task is packed below it.
Server priorities
Server priorities

\[ S^2 \]

\[ S^1* \]

\[ S^1 \]

\[ S^0* \]

\[ S^0 \]

\[ T \]

\[ \tau_1 \]

\[ \tau_2 \]

\[ \tau_3 \]

\[ \tau_4 \]

\[ \tau_5 \]

\[ \tau_6 \]

\[ \tau_7 \]

\[ U(\tau_1) = 0.6 \]

\[ U(\tau_2) = 0.6 \]

\[ U(\tau_3) = 0.3 \]

\[ U(\tau_4) = 0.3 \]

\[ U(\tau_5) = 0.6 \]

\[ U(\tau_6) = 0.3 \]

\[ U(\tau_7) = 0.3 \]

\[ <3.6,6> \]

\[ <3.6,6> \]

\[ <3,10> \]

\[ <4.5,15> \]

\[ <1.8,3> \]

\[ <4.5,15> \]

\[ <9,30> \]
Server priorities

$S^2$

$S^1^*$

$S^1$

$S^0^*$

$S^0$

$T$

$\tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7$

$U(\tau_1) = 0.6$ $<3.6,6>$ $U(\tau_2) = 0.6$ $<3.6,6>$

$U(\tau_3) = 0.3$ $<3,10>$ $U(\tau_4) = 0.3$ $<4.5,15>$ $U(\tau_5) = 0.6$ $<1.8,3>$ $U(\tau_6) = 0.3$ $<4.5,15>$ $U(\tau_7) = 0.3$ $<9,30>$
So, how do we compute the budget of servers at level 1?

- Requires reasoning on the dual problem
- **Target 1:** Providing enough execution time for job completion
- **Target 1:** Ensuring that sporadic releases do not interfere with nominal execution
- Still, it’s a matter of computing the demand from lower-level server
- Now taking into account possible sporadic activations in the future

Our solution is tailored to level 1, but does not scale up
- That’s where optimality is lost

Mathematical formulation in the paper
\[ \tau_3 \text{ releases jobs at } \{3, \ldots \}, \, \tau_4 \text{ at } \{0.3, \ldots \}, \, \tau_5 \text{ at } \{0, 3.5, \ldots \} \]
SPRINT - Example

Diagram showing a state transition graph with states labeled $S^2$, $S^1$, $S^0$, $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, $S_6$, $S_7$, $R$, and transitions labeled $\tau_1$, $\tau_2$, $\tau_3$, $\tau_4$, $\tau_5$, $\tau_6$, $\tau_7$. The diagram includes state probabilities and time intervals.

- $S^2$: States $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, $S_6$, $S_7$ with probabilities $U = 0.2$, $U = 0.4$, $U = 0.6$.
- $S^1$: States $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, $S_6$, $S_7$ with probabilities $U = 0.2$, $U = 0.4$, $U = 0.6$.
- $S^0$: States $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, $S_6$, $S_7$ with probabilities $U = 0.6$.

Times:
- $U(\tau_1) = 0.6$ <3,6,6>
- $U(\tau_2) = 0.6$ <3,6,6>
- $U(\tau_3) = 0.3$ <3,10>
- $U(\tau_4) = 0.3$ <4,5,15>
- $U(\tau_5) = 0.6$ <1,8,3>
- $U(\tau_6) = 0.3$ <4,5,15>
- $U(\tau_7) = 0.3$ <9,30>
SPRINT - Example
SPRINT - Example

The diagram illustrates a Markov decision process (MDP) with states S\(^2\), S\(^1\), S\(^0\), S\(^0^*\), and S\(^1^*\), each with a transition probability of 0.8, 0.4, and 0.6, respectively. The transitions are labeled with the time steps \(\tau_1\) to \(\tau_7\) and their corresponding utilities: U(\tau_1) = 0.6, U(\tau_2) = 0.6, U(\tau_3) = 0.3, U(\tau_4) = 0.3, U(\tau_5) = 0.6, U(\tau_6) = 0.3, and U(\tau_7) = 0.3.

The states are color-coded as follows: S\(^2\) in gray, S\(^1\) in red, S\(^0\) in orange, S\(^0^*\) in green, and S\(^1^*\) in black. The utilities for each state transition are indicated along the edges, with the utilities for the transitions being U = 0.6, U = 0.4, and U = 0.2.

The diagram shows the decision points and the impact of the transitions on the state probabilities over time.
SPRINT - Example

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ \tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7 \]

\[ U(\tau_1) = 0.6, \ \langle 3,6,6 \rangle \]
\[ U(\tau_2) = 0.6, \ \langle 3,6,6 \rangle \]
\[ U(\tau_3) = 0.3, \ \langle 3,10 \rangle \]
\[ U(\tau_4) = 0.3, \ \langle 4,5,15 \rangle \]
\[ U(\tau_5) = 0.6, \ \langle 1,8,3 \rangle \]
\[ U(\tau_6) = 0.3, \ \langle 4,5,15 \rangle \]
\[ U(\tau_7) = 0.3, \ \langle 9,30 \rangle \]
SPRINT - Example

The diagram illustrates a sequence of states and transitions labeled as $S^2$, $S^1$, $S^0$, and $T$. Each state is marked with a probability $U$. For example, $S_1^2$ has $U = 0.2$. The transitions are indicated with arrows, and some transitions are marked with specific values such as $U(\tau_3) = 0.3$. The timeline below the diagram represents the sequence of events with different colors for each event type.
SPRINT - Example

The diagram shows a decision tree for SPRINT with states $S^2$, $S^1$, $S^0$, $S^0^*$, and $S^1^*$, along with their corresponding utilities. The tree branches out from the root $R$ with various states and transitions labeled with utilities. The transitions are marked with tags $\tau_1$ to $\tau_7$ and their respective utilities and intervals.

- $S^2$: States 1, 2, 3, 4, 5.
- $S^1$: States 1, 2, 3, 4, 5.
- $S^0$: States 1, 2, 3, 4, 5.
- $S^0^*$: States 1, 2, 3, 4, 5.
- $S^1^*$: States 1, 2, 3, 4, 5.

The utilities for the transitions are as follows:

- $\tau_1$: $U(\tau_1) = 0.6$, interval $<3,6>$.
- $\tau_2$: $U(\tau_2) = 0.6$, interval $<3,6>$.
- $\tau_3$: $U(\tau_3) = 0.3$, interval $<3,10>$.
- $\tau_4$: $U(\tau_4) = 0.3$, interval $<4,15>$.
- $\tau_5$: $U(\tau_5) = 0.6$, interval $<1,8>$.
- $\tau_6$: $U(\tau_6) = 0.3$, interval $<4,15>$.
- $\tau_7$: $U(\tau_7) = 0.3$, interval $<9,30>$.
SPRINT - Example

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]

\[ S^2 \]

\[ S^1 \]

\[ S^0 \]

\[ T \]
Increasing system utilisation
Increasing number of tasks

- Preemptions per job
- Migrations per job
- Scheduled task sets

**SPRINT - Simulation results /2**
Increasing number of processors

- **SPRINT**
  - Simulation results
  - Increasing number of processors

- **Preemptions per job**
  - **SPRINT (no delay)**
  - **SPRINT (max delay = max period)**
  - **SPRINT (max delay = 10 * max period)**

- **Migrations per job**
  - **SPRINT (no delay)**
  - **SPRINT (max delay = max period)**
  - **SPRINT (max delay = 10 * max period)**

- **U-EDF**
  - Simulation results
  - Increasing number of processors

- **Preemptions per job**
  - **U-EDF (no delay)**
  - **U-EDF (max delay = max period)**
  - **U-EDF (max delay = 10 * max period)**

- **Migrations per job**
  - **U-EDF (no delay)**
  - **U-EDF (max delay = max period)**
  - **U-EDF (max delay = 10 * max period)**
Future work

- Optimality needs further investigation
  - Reasoning on level 1 is not easy, finding a general rule even less
  - Although not optimal, SPRINT schedules the vast majority of task sets

- Comparison needed on a real implementation

- Comparison needed against QPS
  - RUN better than QPS according to QPS paper
  - Preliminary result on a real implementation: it may depend on the specific task set