Fault-Tolerant Online Scheduling Algorithms for CubeSats

Petr Dobiáš[1] **Emmanuel Casseau**[1] **Oliver Sinnen**[2]

[1]Univ Rennes, Inria, CNRS, IRISA, France
[2]PARC Lab, University of Auckland, New Zealand

Bologna, Italy
January 21, 2020

Institut de Recherche en Informatique et Systèmes Aléatoires

Outline

1. CubeSats

2. Model & Algorithm Approaches

3. Results

Layout

1. CubeSats

2. Model & Algorithm Approaches

3. Results

CubeSats [1]

- Small satellites
- ***Several systems***
  - On-board computer
  - Electrical power system
  - Communication system
  - Payload
  - ...
- ***Missions***: Scientific investigations
- ***Problem***: Vulnerable to faults

[1] NASA CubeSat Launch Initiative, *CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers*, 2017, https://www.nasa.gov/sites/default/files/atoms/files/nasa_csli_cubesat_101_508.pdf

Our Aim

- ***Idea***
  - Design fault-tolerant scheduling algorithms
- ***How?***
  - Put all CubeSat processors together on one board
- ***Why?***
  - Reduce space and weight
  - Optimize energy consumption
  - Improve fault tolerance
    - Shared resources: in case of processor failure, a system remains operational
    - Easier protection against faults from radiation
    - Reduction in communication failure rate

Layout

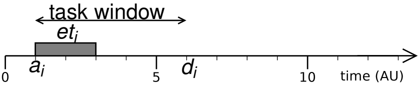
1. CubeSats

2. Model & Algorithm Approaches

3. Results

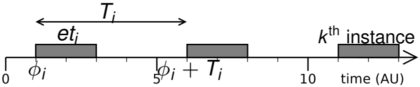
Task & Fault Models

Aperiodic task



- Arrival time a_i
- Execution time et_i
- Deadline d_i
- Task type tt_i

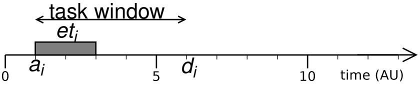
Periodic task



- Phase ϕ_i
- Execution time et_i
- Period T_i = relative deadline
- Task type tt_i

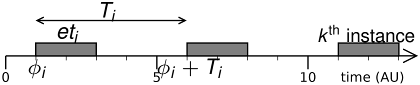
Task & Fault Models

Aperiodic task



- Arrival time a_i
- Execution time et_i
- Deadline d_i
- Task type tt_i

Periodic task



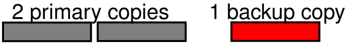
- Phase ϕ_i
- Execution time et_i
- Period T_i = relative deadline
- Task type tt_i

Task type

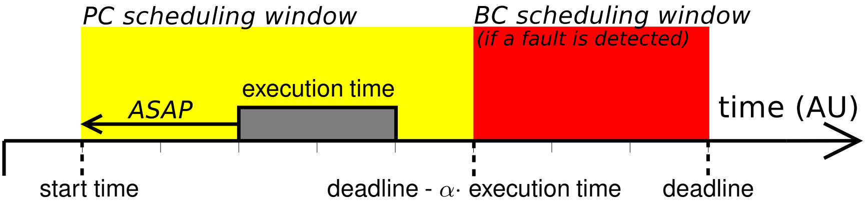
- **Standard tasks**

1 primary copy
1 backup copy

- **Critical tasks**



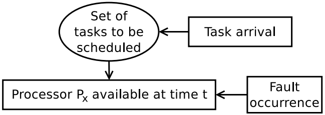
Algorithm to Schedule Tasks



Principle of scheduling task copies ($\alpha \geqslant 1$)

- Task is rejected if it does not meet its deadline

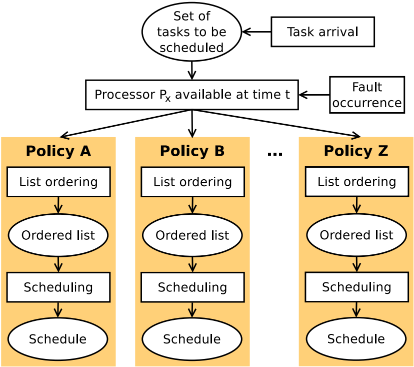
Algorithm 1: Scheduling Tasks as Aperiodic


Set of tasks to be scheduled
Task arrival
Processor P_x available at time t
Fault occurrence

Three main steps

1. **Scheduling triggers**
2. Search for a new schedule
3. Update of task sets

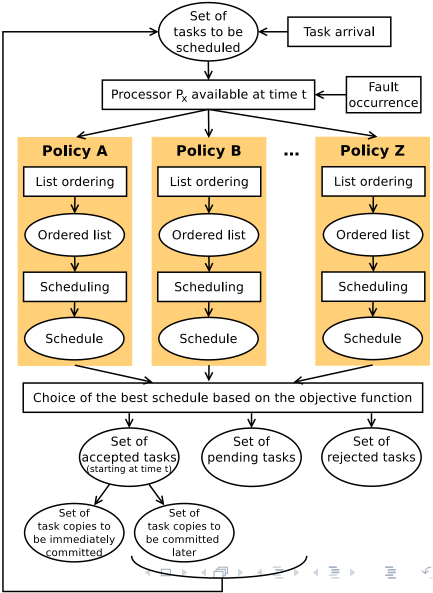
Algorithm 1: Scheduling Tasks as Aperiodic



Three main steps

1. Scheduling triggers
2. **Search for a new schedule**
3. Update of task sets

Algorithm 1: Scheduling Tasks as Aperiodic


Set of tasks to be scheduled
Task arrival
Processor P_x available at time t
Fault occurrence
Policy A
Policy B
...
Policy Z
List ordering
Ordered list
Scheduling
Schedule
Choice of the best schedule based on the objective function
Set of accepted tasks (starting at time t)
Set of pending tasks
Set of rejected tasks
Set of task copies to be immediately committed
Set of task copies to be committed later

Three main steps

1. Scheduling triggers
2. Search for a new schedule
3. **Update of task sets**

Objective function

Minimise the rejection rate subject to correct execution before deadline despite faults

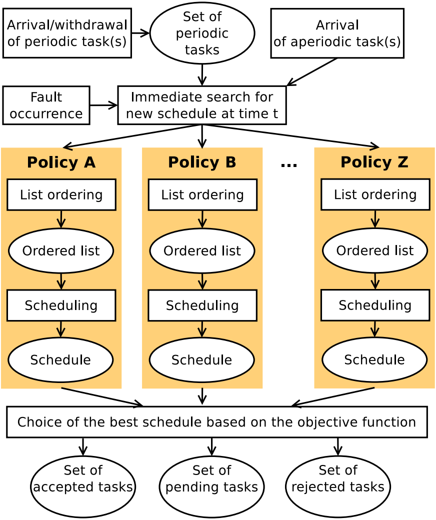
Algorithm 2: Scheduling Tasks as Aperiodic or Periodic

Three main steps

1. Scheduling triggers
2. Search for a new schedule
3. Update of task sets

Objective function

Minimise the rejection rate subject to correct execution before deadline despite faults



Layout

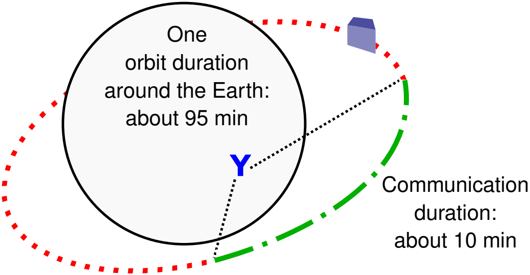
1. CubeSats

2. Model & Algorithm Approaches

3. Results

Experiment Framework

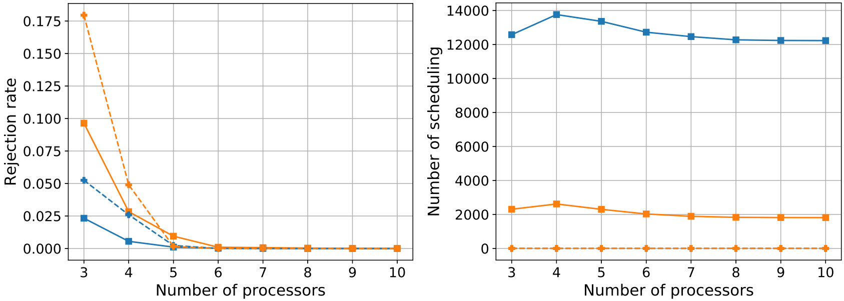
- Data based on tasks from the APSS CubeSat[1]
  - Periodic tasks: *reading/storing data, telemetry, checks*
  - Sporadic tasks: *communication transmission*
  - Aperiodic tasks: *interrupts*



- Two sets of tasks
  - Phase with the communication
  - Phase without the communication

[1] https://space.auckland.ac.nz/auckland-program-for-space-systems-apss/

Comparison of Algorithms 1 and 2



(a) Rejection rate

(b) Number of scheduling searches

Legend

Phase with communication	Phase without communication
━■━ Algorithm 1 (all techniques)	━■━ Algorithm 1 (all techniques)
-✚- Algorithm 2 (all techniques)	-✚- Algorithm 2 (all techniques)

Algorithm 1: Random, Minimum Slack first, Highest ratio of et_i to (d_i-t) first, Lowest ratio of et_i to (d_i-t) first, Longest Execution Time first, Shortest Execution Time first, Earliest Arrival Time first and Earliest Deadline first

Algorithm 2: Random, Minimum Slack first, Longest Execution Time first, Shortest Execution Time first, Earliest Phase first and Rate Monotonic

- Algorithm 1 performs better but at the cost of higher energy consumption

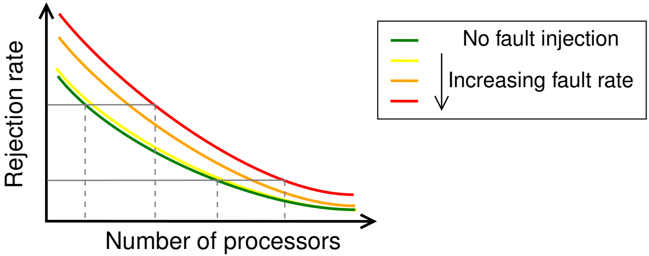
Conclusion & Current Work

Conclusion

- Software solution to provide CubeSats with fault tolerance
- Algorithms adaptable to user demands
- Comparison of different ordering policies

Current work

- Fault injection



Thank you for your attention!

Bibliography I

[1] NASA CubeSat Launch Initiative, *CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers*, 2017. `https://www.nasa.gov/sites/default/files/atoms/files/nasa_csli_cubesat_101_508.pdf`.