# Task Space Searcher (TSS)

By Tyler Bletsch

For CSC714, Prof. Mueller

28 November 2005

Background image adapted from Blatte's fractal wallpapers (http://exoteric.roach.org/)

# Problem

- We know how to check a task set for schedulability if we know all the parameters, but what if we don't?
- For example, we are designing a real-time system, and...
  - We're at the drawing board
    - Want to know what kind of task sets would be most flexible
  - We have tasks, but timing parameters aren't precisely known
    - How tight do we have to get the upper bound?
  - Current task set isn't feasible, but we can optimize tasks further
    - Which tasks are worth spending manpower on?
    - Which tasks don't significantly affect things?
    - How much better do we have to get to become schedulable?

# Approaching the problem

- Allow the user to test *sets* of values for period, WCET, and deadline
- Each of those variables can be:
  - A simple floating-point number, such as “6.5”
  - A set of numbers, such as “{1, 3, 6.5}”
  - A discrete interval of the form “[$a$, $b$, $\delta$]”
    - The set of numbers from $a$ to $b$ stepping by $\delta$.
    - So "[1, 3, 0.5]" = "{1, 1.5, 2, 2.5, 3}"
  - A discrete interval of the form “[$a$, $b$]”
    - Just as above, with $\delta$=1

```
{2,5}; [3,4,0.25]; 5
```

*Expands to:*

```
2; 3   ; 5
2; 3.25; 5
2; 3.5 ; 5
2; 3.75; 5
2; 4   ; 5
5; 3   ; 5
5; 3.25; 5
5; 3.5 ; 5
5; 3.75; 5
5; 4   ; 5
```

# Design

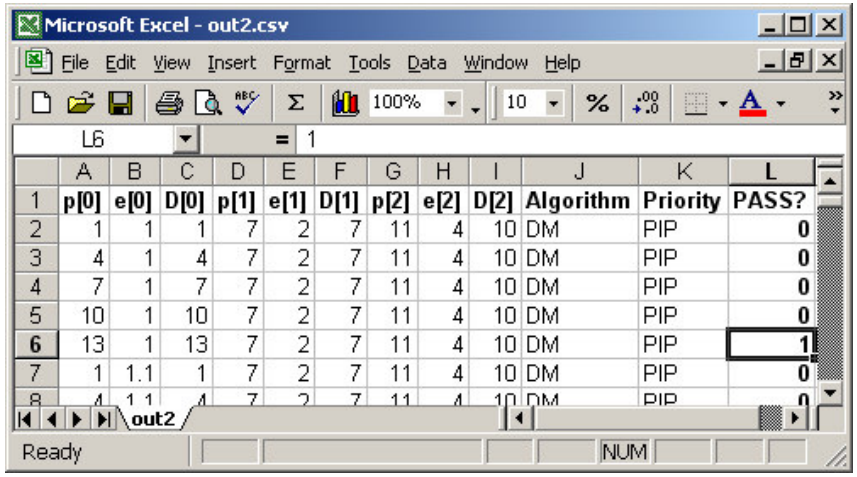
- Read a file indicating the tasks (with wildcards) and the tests to run
- Iterate all possible cases, running appropriate analysis on each
  - Cases do not need to be stored in memory, so memory usage grows linearly with number of tasks
  - Time usage may grow exponentially due to combinative effects of wildcards
- Output possibilities:
  - A simple list of schedulable task sets
  - A verbose analysis printout for each task set
  - A machine-readable Comma-Separated Values (CSV) file detailing the results

# Usage example

- Input file “test.tss”

```
task [1,15,3];[1,5,0.1] # Add first task, which has wildcards
task 7;2;7          # Add a fixed second task
task 11;4;10        # Add a fixed third task

try DM with PIP     # Run the analysis for DM
try EDF with PIP    # Run the analysis for EDF
```




| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | p[0] | e[0] | D[0] | p[1] | e[1] | D[1] | p[2] | e[2] | D[2] | Algorithm | Priority | PASS? |
| 2 | 1 | 1 | 1 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |
| 3 | 4 | 1 | 4 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |
| 4 | 7 | 1 | 7 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |
| 5 | 10 | 1 | 10 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |
| 6 | 13 | 1 | 13 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 1 |
| 7 | 1 | 1.1 | 1 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |
| 8 | 4 | 1.1 | 4 | 7 | 2 | 7 | 11 | 4 | 10 | DM | PIP | 0 |

- Terminal session

```
Task Space Searcher (TSS) by Tyler Bletsch

Usage: tss.pl [-v[#]] [-c<CSVfile>] <TSSFile>

Options:
  -v  Set verbosity to #, or 1 if no number is specified. Defaults to 0. Key:
        0 = Emit successful task sets only
        1 = Print TaskSets & analysis result
        2 = Print all of 1, and include in-depth calculation details
  -c  Output results in Comma-Separated Values (CSV) format to <CSVFile>.
  -p  Just print the cases that would be tested (but don't test them).


DM: (13,1,13) (7,2,7) (11,4,10)
DM: (13,1.1,13) (7,2,7) (11,4,10)
DM: (13,1.2,13) (7,2,7) (11,4,10)
EDF: (4,1,4) (7,2,7) (11,4,10)
EDF: (7,1,7) (7,2,7) (11,4,10)
... and so on ...
EDF: (13,3.9,13) (7,2,7) (11,4,10)
EDF: (13,4,13) (7,2,7) (11,4,10)
```

# Conclusion & Future Work

- We want to expand schedulability analysis to check a number of task sets
  - We allow task parameters to be sets or intervals
  - Test all task sets in the problem space
- Future directions
  - Apply additional reasoning to rule out task sets without performing analysis on them
  - Add support for parallel execution to combat exponential growth in running times

# Any Questions?

My project page is available at:
http://www4.ncsu.edu/~tkbletsc/714/project.html