組込みシステム向け FRP 言語の静的スケジューリングを用いた並列化

櫻井義孝・渡部卓雄

東京工業大学

概要

- 目的:小規模組込みシステム向け FRP 言語の応答性向上
- 既存の小規模組込みシステム向け FRP 言語である Emfrp はシングルスレッドで動作する
  - マルチスレッドが使用可能な環境では計算資源を十分に活用できない
- 本研究では応答性向上のために静的スケジューリングを用いて Emfrp を並列化することを提案する

Emfrp [Sawada2016]

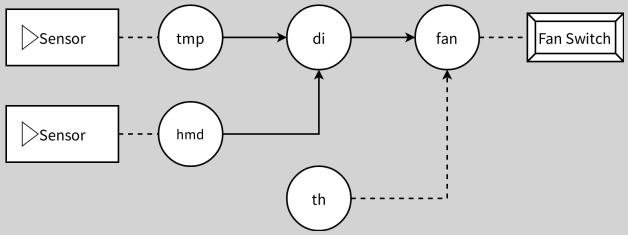
小規模組込みシステム向け FRP 言語

```
module FanController     # 不快指数からファンの
in  tmp: Double,         # オンオフを決定するアプリケーション
    hmd: Double
out di:  Double,
    fan: Bool
use Std

# discomfort index
node di = 0.81*tmp+0.01*hmd*(0.99*tmp-14.3)+46.3

# fan switch
node init[False] fan = di >= th

# threshold
node th = 75.0 + (if fan@last then -0.5 else 0.5)
```

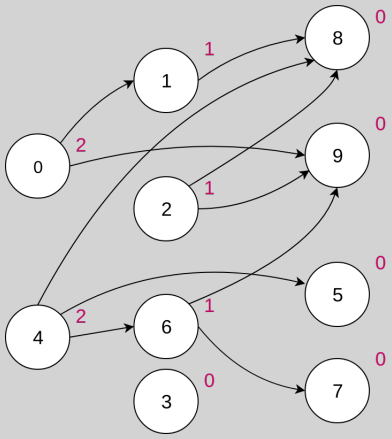


- 時変値をノードとし依存関係を辺とした有向非巡回グラフ (DAG) を構成
- Emfrp の処理系では依存関係に沿ってノードの値を順次更新することを繰り返しリアクティブなシステムを実現している
- この例では例えば tmp→hmd→di→th→fan の順
- しかし，現在の Emfrp の処理系はノードの更新が逐次的に行われているので，ノードの更新を並列に行うことを提案する
- OS がない環境や小規模な Free-RTOS などの環境では OS によるスケジューリングが利用できないことがある
- そこでそのような環境にも対応するため静的にスケジューリングすることを提案する

並列化アルゴリズム

1. 各ノードから sink ノードへの最長距離を計算
2. TODO: sink ノードの説明 (図にどれが sink ノードなのかの説明を入れる)
3. 同じ距離にあるノードを並列に実行
   - OS のスケジューラーを利用できない環境ではどのプロセッサがどのノードを処理するかを指定する必要があるのでその説明を行う

出力ノードまでの距離	2	1	0
同時に処理するノード	0,4	1,2,6	3,5,7,8,9



実験

- 環境
- 対象アプリケーション：ライフゲーム