小規模組み込みシステム向けFRP言語とその自己反映機構

渡部卓雄（東京工業大学 情報理工学院）

マイクロコントローラ等の小規模組み込みシステム向けに設計された関数リアクティブプログラミング(FRP)言語とそのための自己反映計算機構を提案する．提案方式では，FRP言語の特徴である時変値を介した実行系内部へのアクセスを可能にすることで，自己反映計算もリアクティブな操作として実現されている．本機構の導入により，小規模システムでの実行を考慮して静的に実現されている言語の実行系に，ある程度の柔軟性と適応性を与えることが可能になる．

関数リアクティブプログラミング(FRP)

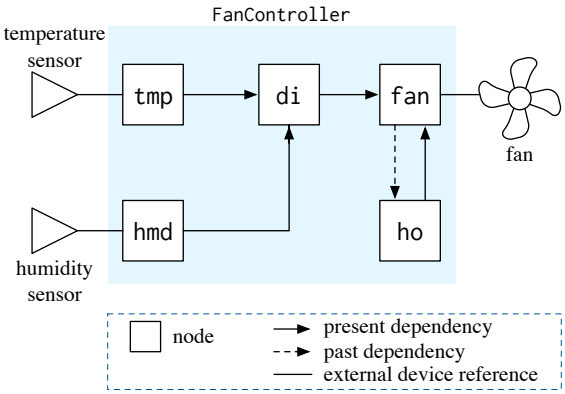
リアクティブプログラミングは，時間と共に変化する値を表す時変値(time-varying value)やイベントストリーム等の抽象化機構を用いて入力やそれらに依存する値を表現し，リアクティブシステムの効果的な記述を支援するプログラミングパラダイムである．特に関数リアクティブプログラミング(FRP)は，関数プログラミングに時変値を導入することでリアクティブシステムの宣言的な記述を可能にする．

時変値はシグナルとも呼ばれる．型Tの時変値の型 Signal T は，概念上時刻から型Tへの関数の型 Time → T で表すことができる．しかし時刻を表す値をプログラム中に明示的に導入すると時間・空間漏洩等の不都合が生じるため，通常はTimeを隠蔽した形で時変値を表す．そのための手法としてはシグナル関数の導入等がある．本研究で提案する言語では，時変値を組込み型とすることでTimeを隠蔽している．加えて，時変値を一級データとはせず，かつ時変値の時変値は扱わないといった制約を設けることで，表現力は失わずにリソース制約の厳しい小規模組込みシステムでの実現を可能にしている．

```
module FanController  # module name
in  tmp : Float,      # temperature sensor
    hmd : Float       # humidity sensor
out fan : Bool        # fan switch
use Std               # standard library

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

# fan switch
node init[False] fan = di >= 75.0 + ho

# hysteresis offset
node ho = if fan@last then -0.5 else 0.5
```

EmFrpによるファンのコントローラ

FRP言語の実行方式



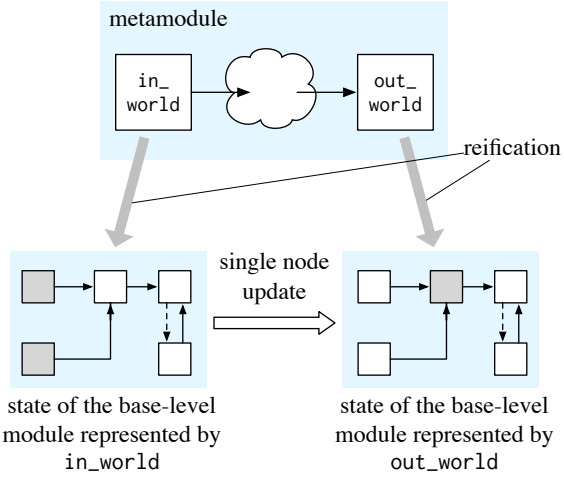
FRP言語によるプログラムは，時変値をノードとし，それらの依存関係を辺とした有向非巡回グラフ(DAG)で表現できる．プログラムの実行は，依存関係に沿ってノードの値を順次更新することの繰り返しとして実現される．左図の場合，例えば tmp, hmd, di, ho, fan の順にノードの値を更新する．この実行方式はpush型と呼ばれる．これに対してpull型の実行方式では，出力に相当するノード（左図の例ではfan）が依存するノードに更新要求を出し，それが入力に相当するノード（例ではtmpとhmd）に至るまで順次伝搬する．

push型の実行方式は少ないリソースで実現可能であり，かつインターバルタイマーによる周期的タスクの表現などに適している．その一方で，pull型の実行では生じないような無駄な計算（不要なノードの更新）が発生することがある．本研究では，push型の実行系に自己反映計算機構を導入することで，より柔軟な実行を可能にする手法を提案する．

FRP言語のための自己反映機構

自己反映計算（リフレクション）は，計算システムが自分自身の構造や実行方式について観測や変更を行なえるようにする技術である．いくつかのオブジェクト指向言語における，クラスやメソッドをオブジェクトとしてプログラム中で操作できる仕組みも自己反映計算の一種である．

本研究では，FRPによるプログラムを表現するDAGを時変値として表現するFRPのプログラム（メタモジュール）を導入し，それを用いて実行のカスタマイズを可能にする手法を提案する．右図上部がメタモジュールであり，そのin_worldおよびout_worldというノードによって，下部にあるアプリケーションプログラムの状態を表現している．in_worldからout_worldを得る計算もFRPのプログラムとして表現される．加えて，アプリケーションから時変値を介してメタモジュールの動作に影響を与えるプログラムの実現も可能になる．つまり自己反映計算を宣言的かつリアクティブに行なうことができる．



Takuo Watanabe & Kensuke Sawada, "Towards Reflection in an FRP Language for Small-Scale Embedded Systems", LASSY 2017, ACM, 2017.