メタ計算を用いた

Continuation based C の検証手法

比嘉 健太

プログラミング言語とソフトウェアの信頼性

- 信頼性の高いソフトウェアを提供したい
- ソフトウェアの仕様を検証するには二つの手法がある
  - ✦ プログラムの取り得る状態を列挙し、
    仕様に背く状態が無いかを調べるモデル検査
  - ✦ プログラムの性質を直接証明する定理証明
- モデル検査も証明も行ないやすい言語として Continuaion based C 言語を開発している

二つのアプローチを用いた検証

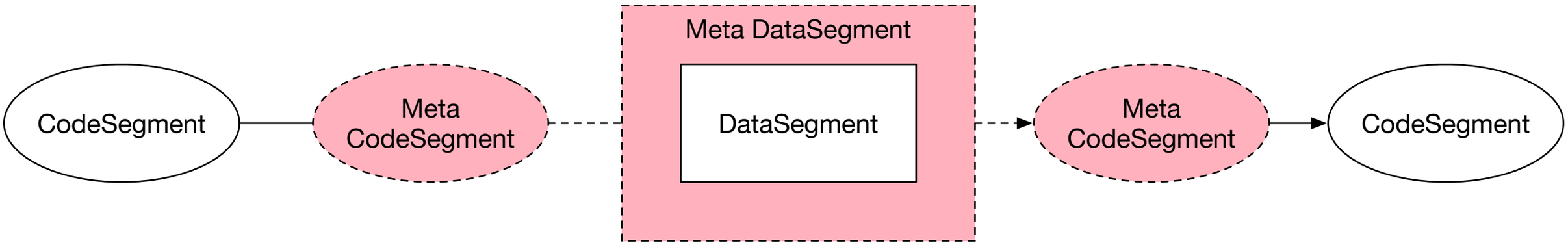
- モデル検査的アプローチ
  - ✦ メタ計算ライブラリ aksha による網羅的な実行
  - ✦ 非破壊赤黒木の仕様定義と検証
- 定理証明的なアプローチ
  - ✦ 証明支援系言語 Agda 上で CbC を定義する
  - ✦ Agda を通して CbC の形式的な定義を得る
  - ✦ SingleLinkedStack の性質の証明

Continuation based C

- 当研究室で開発しているプログラミング言語
- アセンブラとC言語の中間のような言語
- CodeSegment と DataSegment という単位でプログラミングを行なう
- CodeSegment どうしを繋げてプログラムを構成

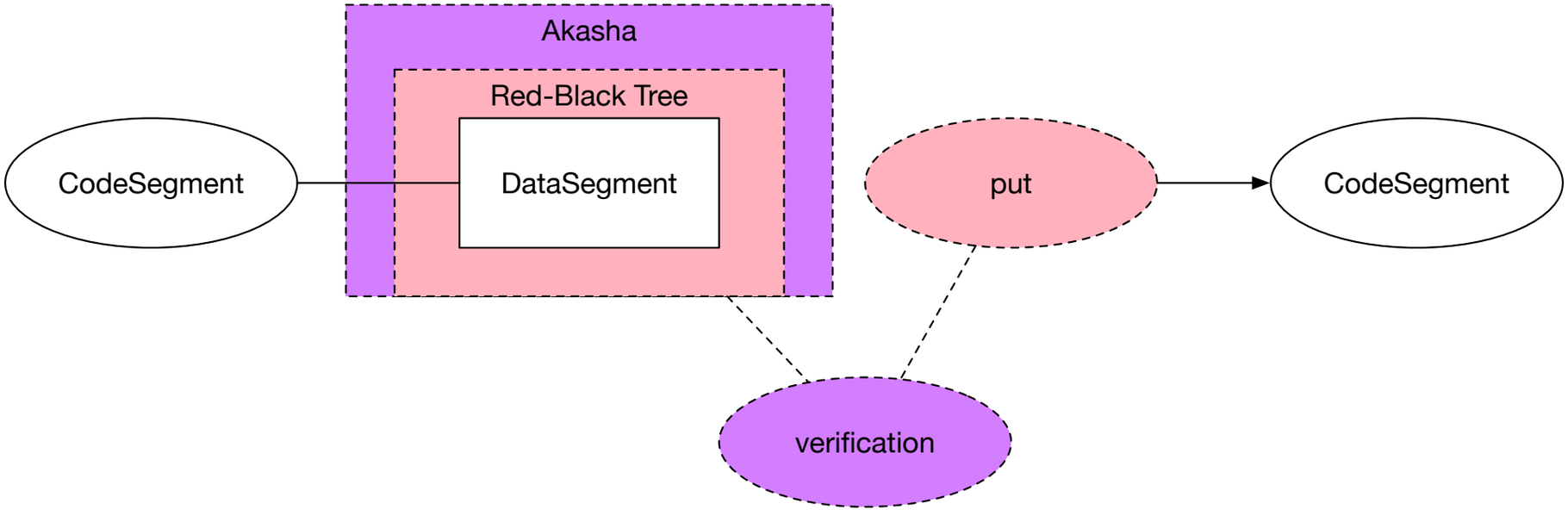
CbC とメタ計算

- CbC では通常の計算とメタ計算を分離
- 通常の計算を拡張することでユーザの負担を減らしつつ複雑なプログラムを構成可能にする
  - ✦ 例外を考慮していない計算を拡張して安全にするなど
- 接続部分に計算を挟むことで実現



メタ計算とモデル計算

- 網羅的に実行するように接続部分を上書き
- 検査対象として非破壊赤黒木の性質を検査
- 「木をルートから辿った際に最も長い経路は最も短い経路の高々2倍に収まる」
- 要素数13個まで仕様を保証



定理証明とAgda

- 任意の回数だけ操作しても仕様を保証したい
- プログラムを直接証明してしまう
- CbC で CbC を証明したいが現状は行なえない
- 証明支援系 Agda 上で CodeSegment と DataSegment を定義する
- CbC の形式的な定義を Agda 上での定義から得る

Agda による CbC プログラムの証明

- CodeSegment は関数型
- DataSegment はレコード型
- メタ計算は部分型
- Agda 上で SingleLinkedStack の操作の仕様を保証
- 「任意の回数だけ値を積んで同じだけ取り出すとスタックは変化しない」

n-push-pop-type n cn ce st = M.exec (M.csComp (n-pop n) (n-push n)) meta ≡ meta
-- goto (pop*n . push*n) mds ≡ mds

まとめ

- 検証しやすい言語 CbC に対する二つのアプローチ
- 一つは状態の列挙による仕様の検査
  - ✦ 限定的な木のサイズに対して仕様を保証
  - ✦ 有界モデル検査器 CBMC より大きな範囲を検査
- 一つは Agda を用いた CbC プログラムの証明
  - ✦ 任意の回数の操作に対する仕様を保証