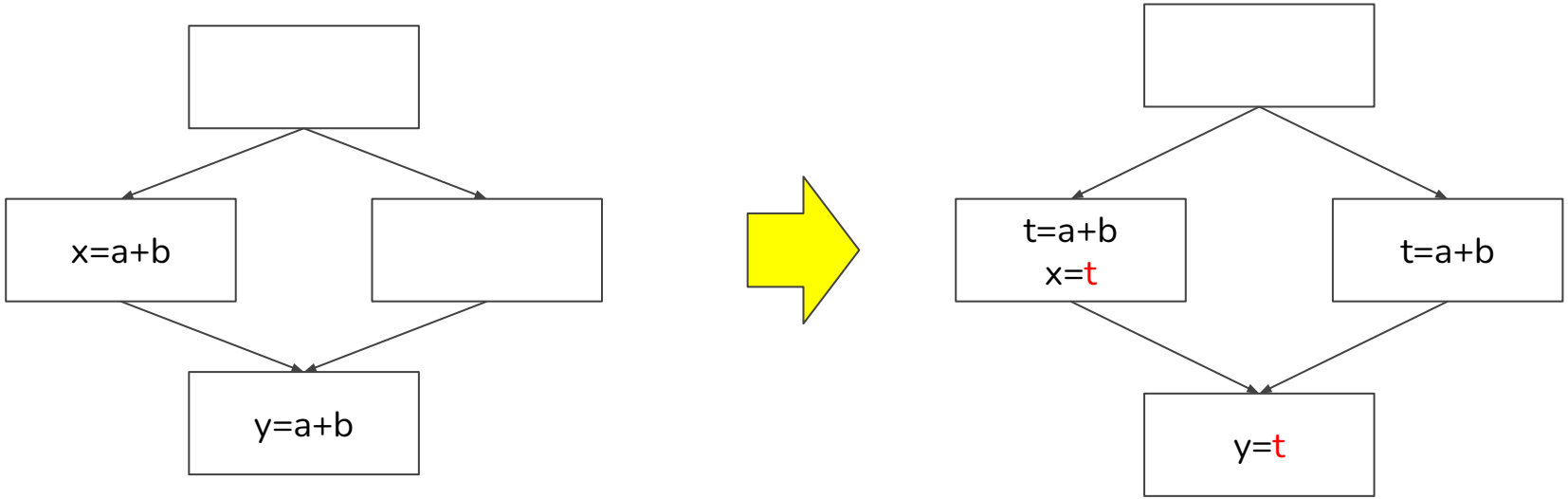
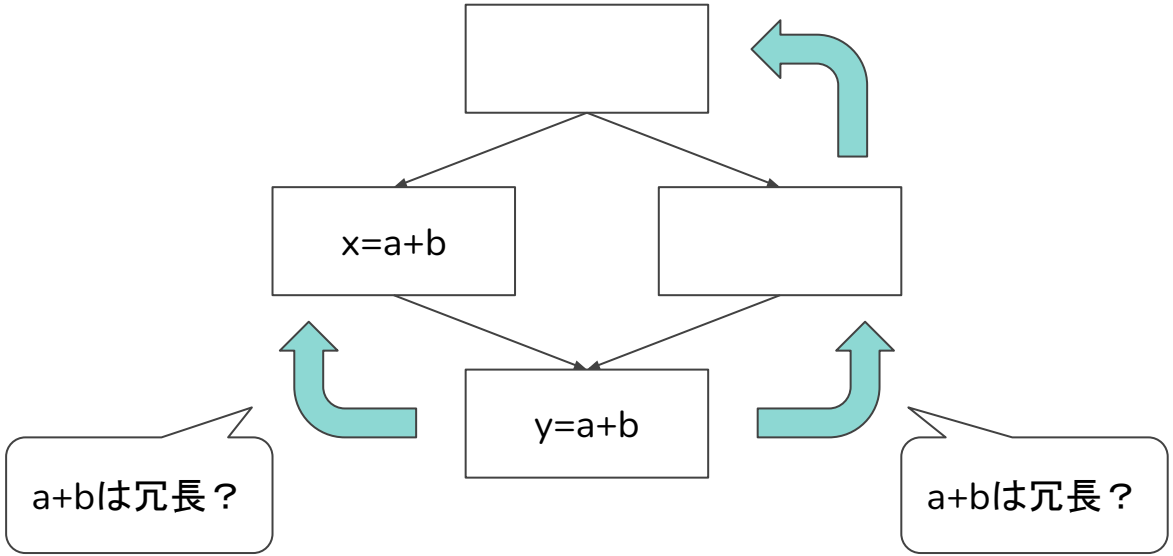
# 実行時情報を利用した要求駆動型部分冗長除去

植村拓凪、澄川靖信
拓殖大学

# 部分冗長除去（PRE）

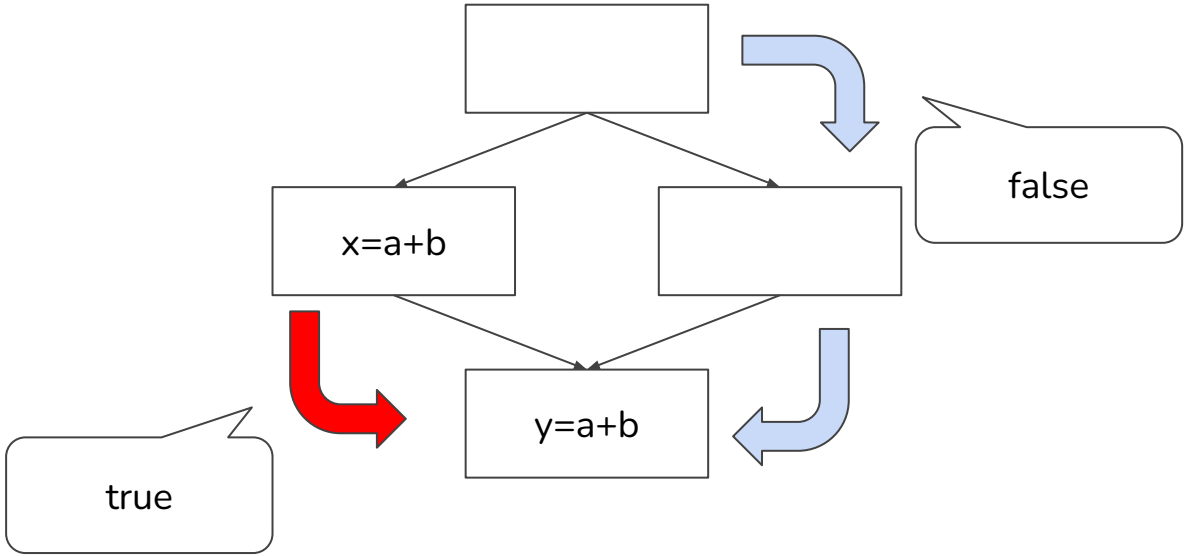


# 要求駆動型 PRE（DDPRE）

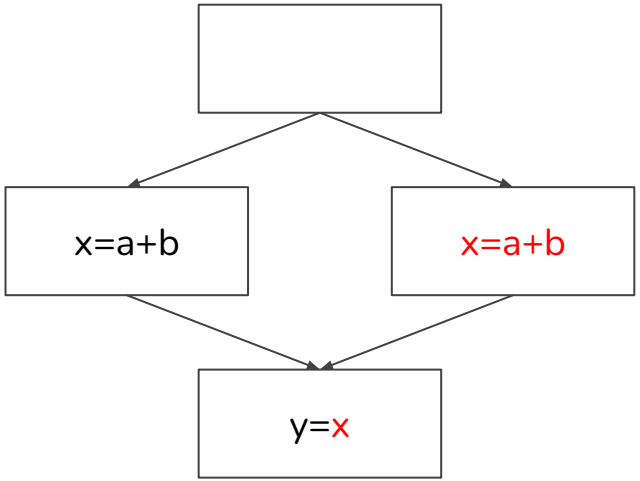



[1]: 滝本宗宏、質問伝播に基づく投機的部分冗長除去、Trans. Prog., Vol.2, No.5, 15-27, 2010.

# 要求駆動型 PRE（DDPRE）



# 要求駆動型 PRE（DDPRE）



# 既存研究の限界

・質問伝播はCFGをトポロジカルソート順序で訪問する

・しかし、プログラムの実行回数は基本ブロックごとに異なる

→ **PREの効果が低い箇所も解析しないといけない**

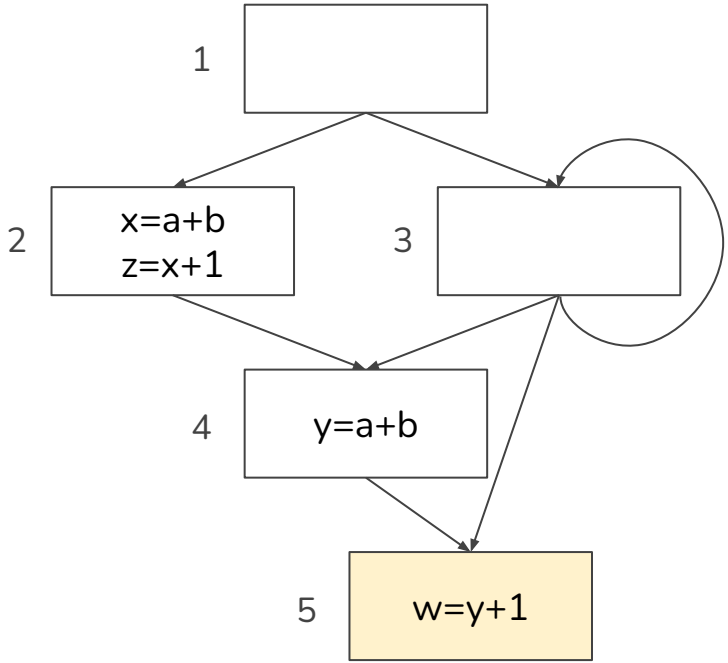
# 本研究の目的

・実行回数が多いところのみにDDPREを適用する

→ 解析対象を上位k%に絞ることができる

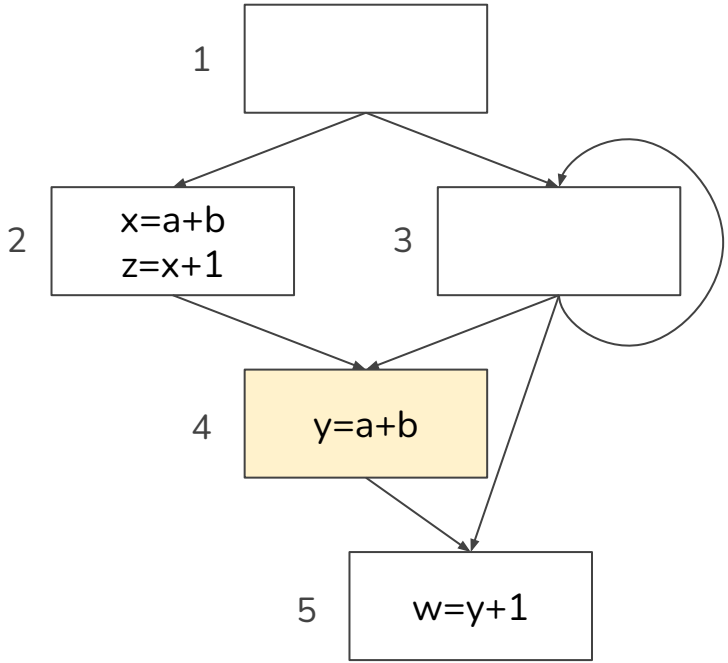
・各基本ブロックの実行時情報（Profile）を利用したD**DPRE**（以降、PDPREと呼ぶ）を提案

# PDPRE｜訪問順



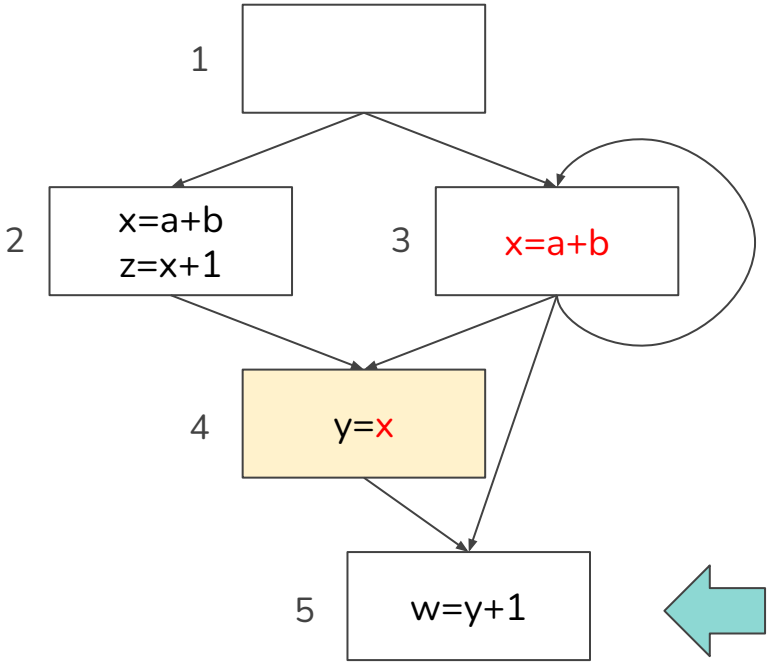

| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜訪問順

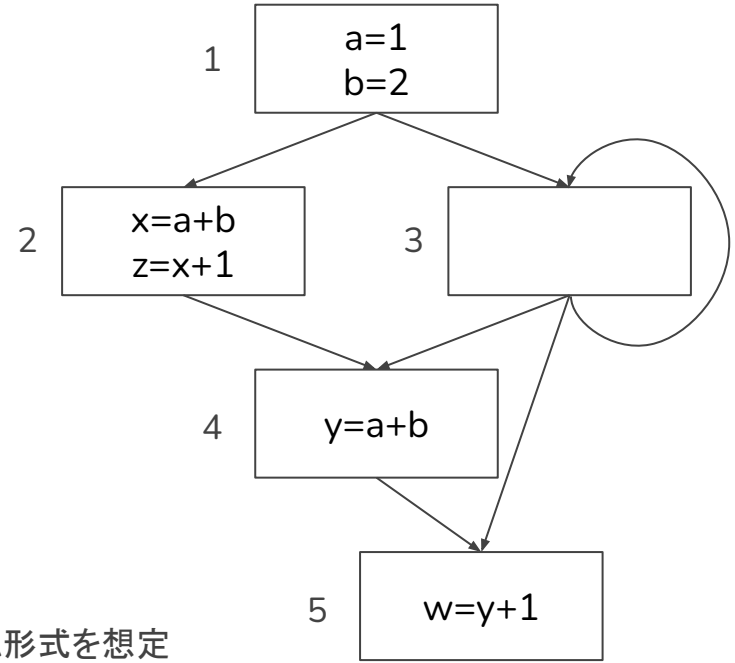



| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜訪問順



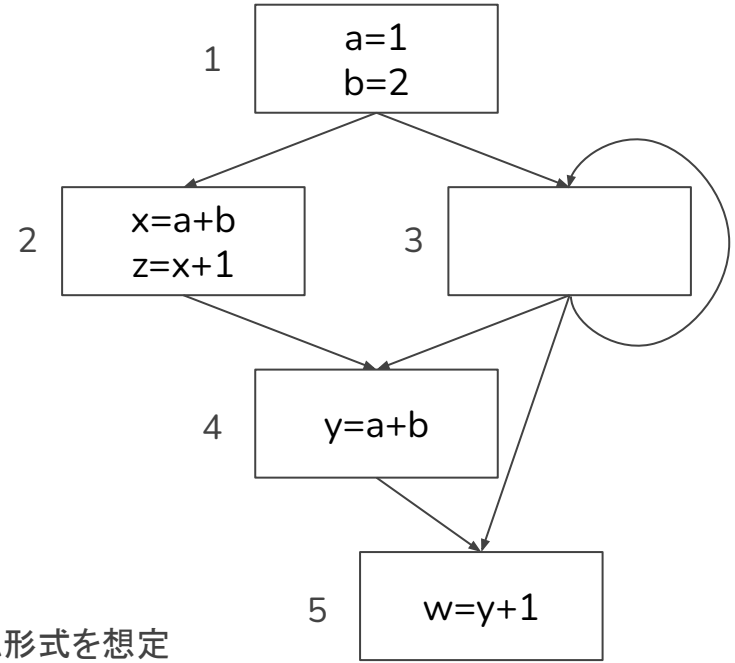
# 大域値番号付け（GVN）




※ 本来はSSA形式を想定

| 項 | 値番号 |
|---|---|
| a=1 | {1} |
| b=2 | {2} |
| x=a+b | {3} |
| z=x+1 | {4} |
| y=a+b | {3} |
| w=y+1 | {4} |

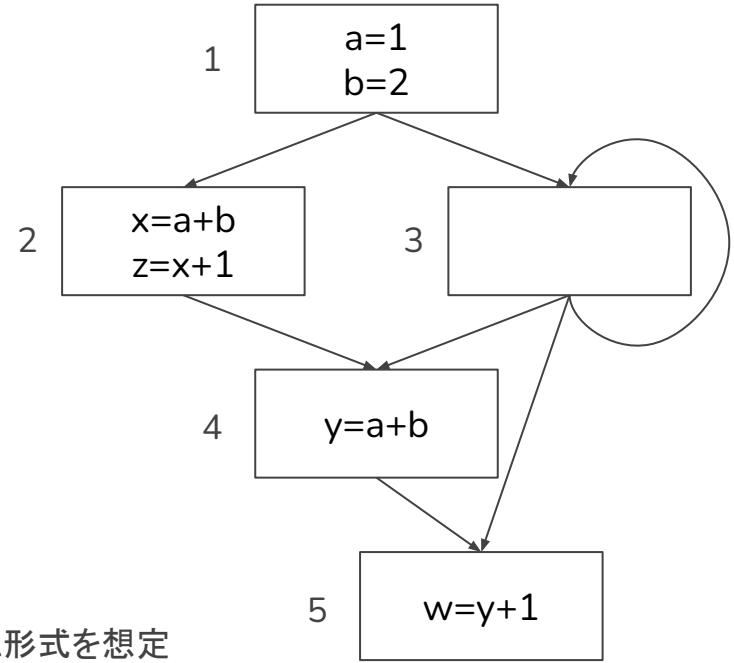
# 大域値番号付け（GVN）




※ 本来はSSA形式を想定

| 項 | 値番号 |
|---|---|
| a=1 | {1} |
| b=2 | {2} |
| x=a+b | ⇒{1}+{2} |
| z=x+1 | {4} |
| y=a+b | ⇒{1}+{2} |
| w=y+1 | {4} |

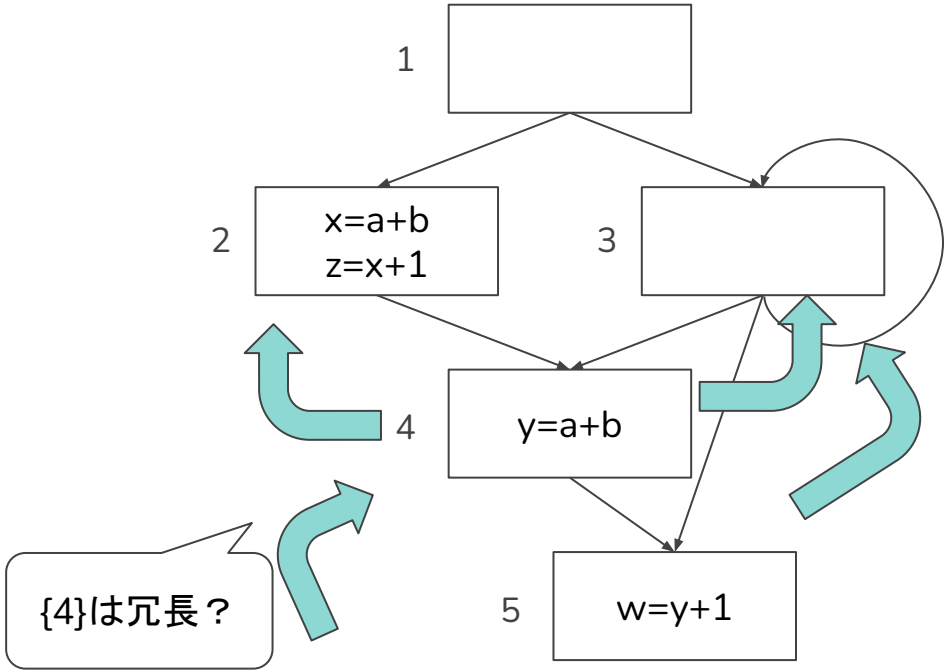
# 大域値番号付け（GVN）



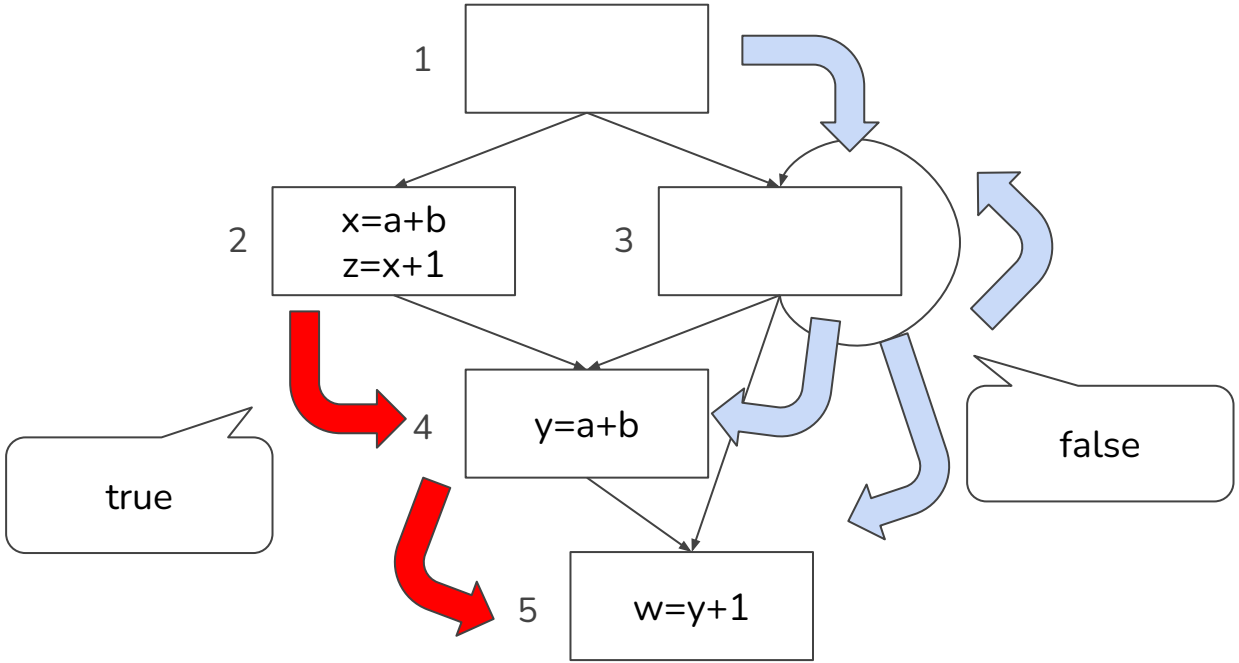

※ 本来はSSA形式を想定

| 項 | 値番号 |
|---|---|
| a=1 | {1} |
| b=2 | {2} |
| x=a+b | {3} |
| z=x+1 | ⇒{3}+{1} |
| y=a+b | {3} |
| w=y+1 | ⇒{3}+{1} |

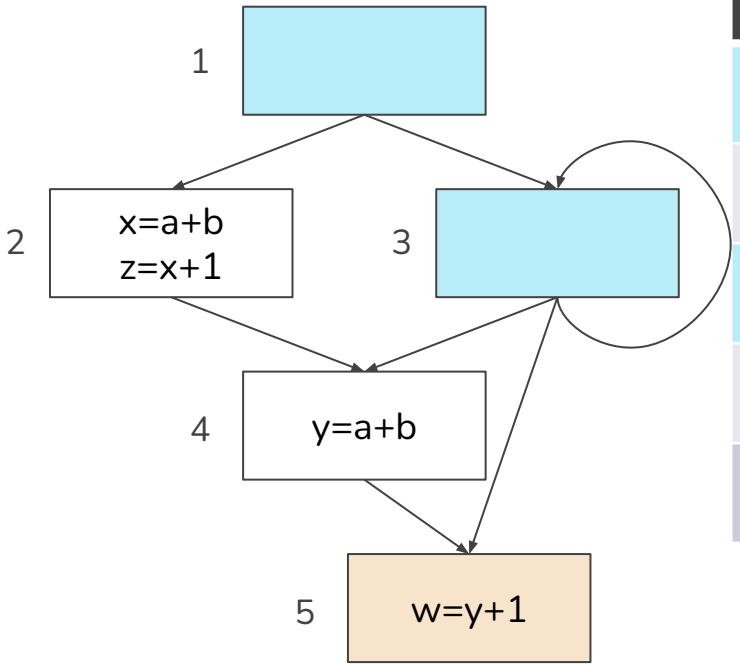
# PDPREの質問伝播



# PDPREの質問伝播

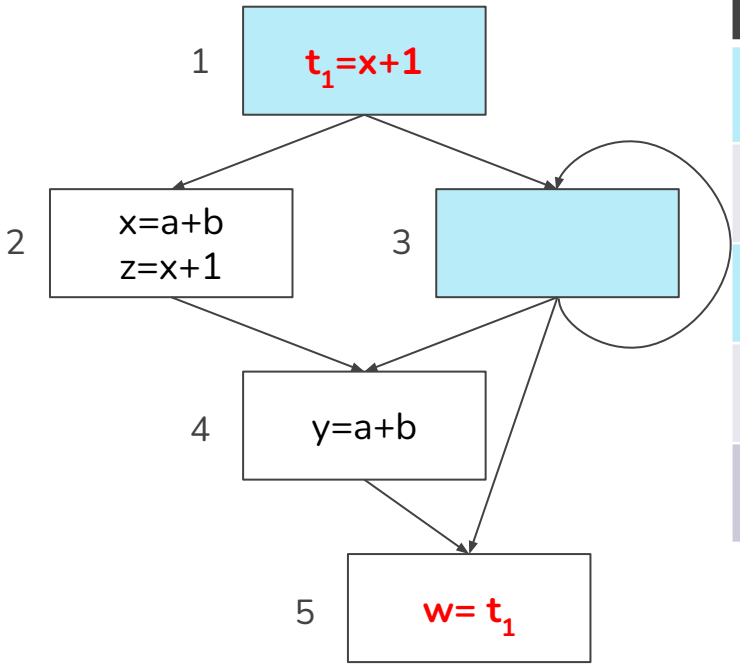


# PDPRE｜挿入手順



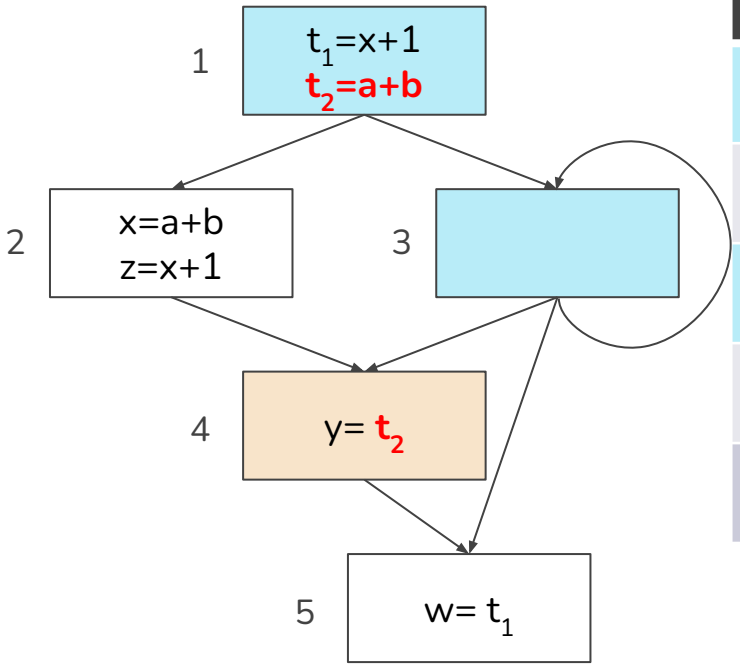

| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜挿入手順



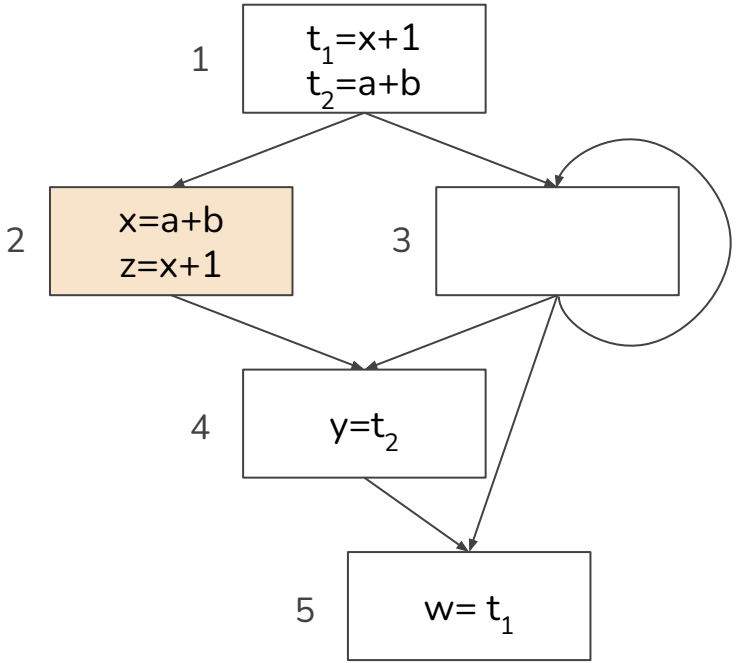

| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜挿入手順



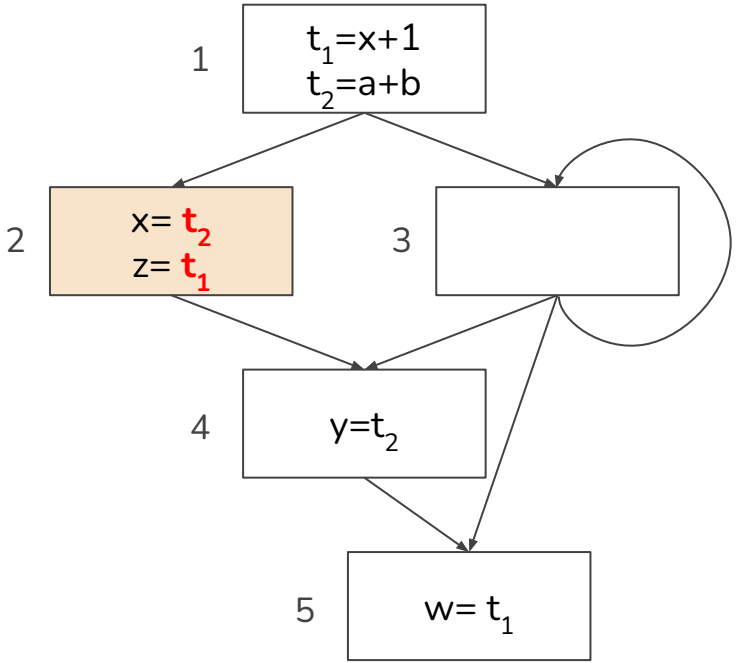

| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜挿入手順




| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# PDPRE｜挿入手順




| CFG節 | 実行回数 |
|---|---|
| 1 | 50 |
| 2 | 30 |
| 3 | 120 |
| 4 | 40 |
| 5 | 50 |

# 実験

・実験環境

・OS: Ubuntu

・Intel Corei7-8700K 3.70GHz CPU

・ベンチマーク：SPEC CPU 2000（10回実行した平均値を表示）

・比較対象：

・PREQP[1]: ループ不変式のみ投機的にループ外へ移動する

・LDPRE[2]：質問伝播の解空間を束で定義したDDPRE

[2]: Y. Yanase and Y. Sumikawa, Lazy Demand-driven Partial Redundancy Elimination, Journal of Information Processing, Vol. 31, pp. 459–468, 2023.

# 実験項目

**RQ1**: PDPREを適用する**上位k%のkとして良いものはどれ**か

= 目的コードの実行時間が短いほど良い。

**RQ2**: DDPREのうち、**解析時間と実行時間が良いのはどれ**か

# RQ1 | 上位0~100%の実行時間（単位：秒）、赤字は最良結果

| | 100 | 90 | 80 | 70 | 60 | 50 | 40 | 30 | 20 | 10 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| gzip | 76.7 | 75 | 75.2 | 77.1 | **74.4** | 75.5 | 75.9 | 76.5 | 74.9 | **74.4** | 74.7 |
| vpr | 51.2 | 52.1 | 52.3 | 51.4 | 51.7 | 51.8 | 51.7 | 52.5 | 51.6 | 51.2 | **51.1** |
| mcf | 23.9 | 24 | 23.8 | 23.9 | 23.6 | 23.7 | 24.1 | 24 | **23.5** | 23.9 | 23.9 |
| parser | 116 | 116.4 | 116.5 | 116.8 | 115.9 | 115.3 | 115 | 114.9 | 115.3 | **114** | **114** |
| gap | 59.9 | 59.9 | 60.1 | 60.3 | 60 | 58.7 | 58.7 | **58.4** | 59.8 | 60.2 | 59.9 |
| bzip2 | 59.9 | 60 | 59.5 | 57.6 | 57.2 | 57.5 | 57 | **56.2** | 57.3 | 57.3 | 57.7 |
| twolf | 84 | 85.4 | 83.6 | 85 | 85.5 | 85 | 84.8 | **83.2** | 84.9 | 83.8 | 84.7 |
| art | 21 | 21 | 21 | 20.5 | 20.4 | **20** | 21.2 | 21.2 | 21.2 | 21 | 22 |
| equake | 33 | 32.7 | 32.2 | 33.2 | 32.5 | 32.4 | 32.3 | 32.3 | 32.3 | 32.4 | **32** |
| ammp | 82.5 | 81.1 | **79.4** | 82.1 | 81.6 | 81.2 | 81.1 | 81.3 | 81.8 | 80.9 | 80.7 |

# 議論

## なぜkの値が増えると実行時間が増加傾向になる？

・部分冗長を全冗長にした数が多くなるにつれてレジスタスピルが多く発生した
　→除去する数が多くなればなるほど挿入する命令数も増える

除去数の多さとスピルの発生が実行時間に影響していると予測　⇒　解析対象を絞ったほうがいい

# RQ2 | DDPREのうち、実行時間が良いのはどれか

| | PREQP | LDPRE | PDPRE |
|---|---|---|---|
| gzip | **75.3** | 77.2 | 76.5 |
| vpr | **50.8** | 53.2 | 52.5 |
| mcf | 24.4 | 24.1 | **24** |
| parser | 115 | 116.5 | **114.9** |
| gap | 59.1 | 59.4 | **58.4** |
| bzip2 | 58.2 | 58.4 | **56.2** |
| twolf | 86.3 | **82.5** | 83.2 |
| art | **20.8** | 24.8 | 21.2 |
| equake | 33.8 | 40.5 | **32.3** |
| ammp | **80.9** | 81.4 | 81.3 |

# RQ2 | DDPREのうち、解析時間が良いのはどれか

| | A. PREQP | B. LDPRE | C. **PDPRE** | (A-C)/A | (B-C)/B |
|---|---|---|---|---|---|
| gzip | 773 | 893 | **398** | 48.5% | 42.9% |
| vpr | 1,509 | 1,798 | **939** | 37.8% | 67.9% |
| mcf | 351 | 414 | **185** | 47.3% | 53.0% |
| parser | 1,366 | 1,615 | **640** | 53.1% | 72.2% |
| gap | 8,194 | 10,057 | **3,225** | 60.6% | 90.3% |
| bzip2 | 295 | 462 | **124** | 58.0% | 79.9% |
| twolf | 5,775 | 6,914 | **2,139** | 63.0% | 88.7% |
| art | 131 | 126 | **91** | 30.5% | 55.0% |
| equake | 367 | 432 | **146** | 60.2% | 79.1% |
| ammp | 2,454 | 2,923 | **1,189** | 51.5% | 70.3% |

# まとめ

・実行時情報を利用したDDPREを提案した

・PDPREは上位30%の適用が最良

・PDPREは他2つのDDPREよりも実行時間，解析時間の効率が良い

**今後の課題**

・PREでは除去できない冗長性を除去するための要求駆動型解析を利用する手法に実行時情報を利用できるように拡張する．