A Reactive Strategy for High-Level Consistency During Search

R.J. Woodward\textsuperscript{1,2} B.Y. Choueiry\textsuperscript{1} Christian Bessiere\textsuperscript{2}

\textsuperscript{1}Constraint Systems Laboratory, University of Nebraska-Lincoln
\textsuperscript{2}CNRS, University of Montpellier

Acknowledgements
• Experiments conducted at UNL’s Holland Computing Center
• NSF Grant RI-111795, RI-1619344
• NSF GRF and Chateaubriand Fellowship
• ANR Demograph (ANR-16-CE40-0028) and Contredo (ANR-16-CE33-024)
Context

• Solving a Constraint Satisfaction Problem (CSP)
  – Conditioning: Backtrack search
  – Inference: Enforcing consistency
    • Consistency properties (e.g., GAC)
    • Constraint propagation algorithms

• Consistency during search
  – Constraint Programming solvers: GAC or weaker
  – CSP research: GAC or stronger

• Our focus: Higher-level consistency (HLC)
Lesson and Problem

- Maintaining consistency during search
  - Enforced at each variable instantiation
  - Prunes subtrees, reduces search space
- Stronger consistency
  - Filters more subtrees
  - But is costlier to enforce
Decide **when**, **where**, and **how much** HLC to enforce during search

- **Always HLC**
  - **When?**
  - **Where?**
  - **How much?**
- **Entire CSP**
  - **When?**
  - **Where?**
  - **How much?**
- **Until fixpoint**
  - **When?**
  - **Where?**
  - **How much?**

- **Always GAC**
  - **When?**
  - **Where?**
  - **How much?**
- **One variable**
  - **When?**
  - **Where?**
  - **How much?**
- **Stop early**
  - **When?**
  - **Where?**
  - **How much?**
Our solution

1. **When: PrePeak**
   - Monitors search performance
   - When search starts thrashing, triggers an HLC
   - Then, conservatively reverts to GAC

2. **How much**
   - Monitor propagation and interrupt before fixpoint

\[
\text{PrePeak}^+ = \text{PrePeak} + \text{‘How Much’}
\]
PREPEAK examines #BT per depth

- Trigger HLC when #BT reaches a threshold $\theta$
- If HLC yields
  1. Wipeout (highly effective), reduce $\theta$
  2. Some filtering, increase $\theta$ a little
  3. No filtering (HLC is wasteful), increase $\theta$ a lot
- In the last 2 cases, reset #BT per depth to 0
  Thus, return to GAC
Empirical Evaluations

Using $\text{dom/deg}$
Thank You

Goal state

Possible start states

Exhibition

Demo booths

Posters

Coffee

Coffee

Coffee
Thank You

Questions & Comments

Please stop by the poster #51
Visualization of Benefit

GAC
CPU Time: 140.81 sec.
#NV: 3,978,074
#BT: 3,348,330

APOAC
CPU Time: 23.77 sec.
#NV: 59,181
#BT: 53,212
#Calls POAC: 11,142

pseudo-aim-200-1-6-4, dom/wdeg
Visualization of Benefit

GAC
CPU Time: 140.81 sec.
#NV: 3,978,074
#BT: 3,348,330

APOAC
CPU Time: 23.77 sec.
#NV: 59,181
#BT: 53,212
#Calls POAC: 11,142

pseudo-aim-200-1-6-4, dom/wdeg
Visualization of Benefit

GAC
CPU Time: 140.81 sec.
#NV: 3,978,074
#BT: 3,348,330

APOAC
CPU Time: 23.77 sec.
#NV: 59,181
#BT: 53,212
#Calls POAC: 11,142

PREPEAK
CPU Time: 12.06 sec.
#NV: 284,289
#BT: 238,833
#Calls POAC: 228
A Reactive Strategy for HLC

- Keep track of $btcount[\cdot]$, number of backtrack during search
- When $btcount[\cdot]$ reaches a given threshold $\theta$
  - Enforce GAC then HLC as long as HLC yields domain wipeout for all values in domain of current variable
  - If backtrack, reduce threshold and keep enforcing HLC
  - If HLC finds a consistent value, reset $btcount[\cdot]$, increase threshold a little
  - If GAC finds a consistent value, reset $btcount[\cdot]$, increase threshold a lot

- Geometric laws to update threshold
  - Wipeout: $\theta_{k+1}^{bt} \leftarrow r_w \cdot \theta_k^{bt}, r_w = 1.2^{-1}$
  - Filtering: $\theta_{k+1}^{bt} \leftarrow r_{f} \cdot \theta_k^{bt}, r_f = 1.2^2$
  - No filtering: $\theta_{k+1}^{bt} \leftarrow r_n \cdot \theta_k^{bt}, r_n = 1.2^3$