A Fast Cutting-Plane Algorithm for Optimal Coalescing

Daniel Grund[1] Sebastian Hack[2]

[1]Saarland University

[2]ENS Lyon

Joint work started at the University of Karlsruhe

Compiler Construction 2007

Outline

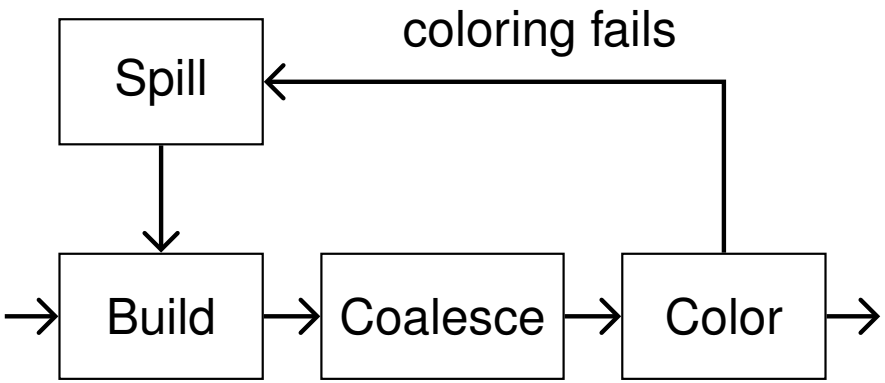
Coalescing

- Backend optimization removing useless copy instructions
- Reduces code size and improves performance

```
mov ebx, [esi]          mov eax, [esi]
add ebx, ebx      ⇒     add eax, eax
mov eax, ebx
mul ecx                 mul ecx
```

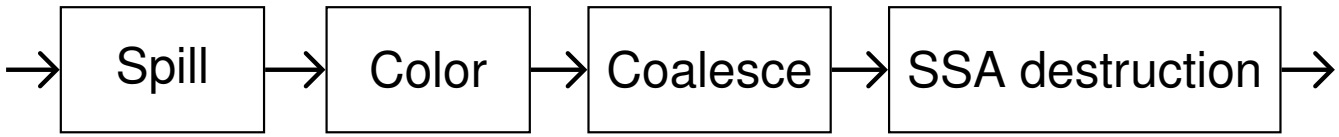
- Needs information about assigned registers
- Therefore subtask of register allocation

Coalescing History in a Nutshell



- Problem: Coalescing influences colorability
- First heuristic ignored negative effects (aggressive)
- Later approaches restricted coalescing (conservative/iterated)
- State of the art has undo-capabilities (optimistic)

Static Single Assignment Helps



- Each subtask needs to be processed only once
- SSA interference graphs are chordal
  - which ensures *register pressure* = *chromatic number of IG*
- Coalescing is isolated subordinate optimization
  - must not provoke further spills

Importance of Coalescing in SSA-RA

- Register constraints destroy equality
  register pressure $=$ *chromatic number*
- Live-range splitting can re-establish this property
- Actual problem/complexity is shifted to coalescing

$\Rightarrow$ Extra stress on coalescing

Contributions / Our Setting

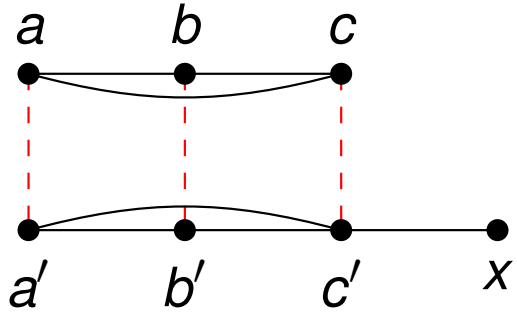
- Problem modeling
  - augmented interference graph
  - affinity edges represent costs of ‘copies’
- Affinity-edge sources
  - register constraints
  - Φ-functions
- Optimal solutions by *optimized* $0/1$ linear program
- Assessment of heuristics

Register Constraint Handling

$$\begin{aligned} x &= op(a,b) \\ d &= c + x \end{aligned}$$

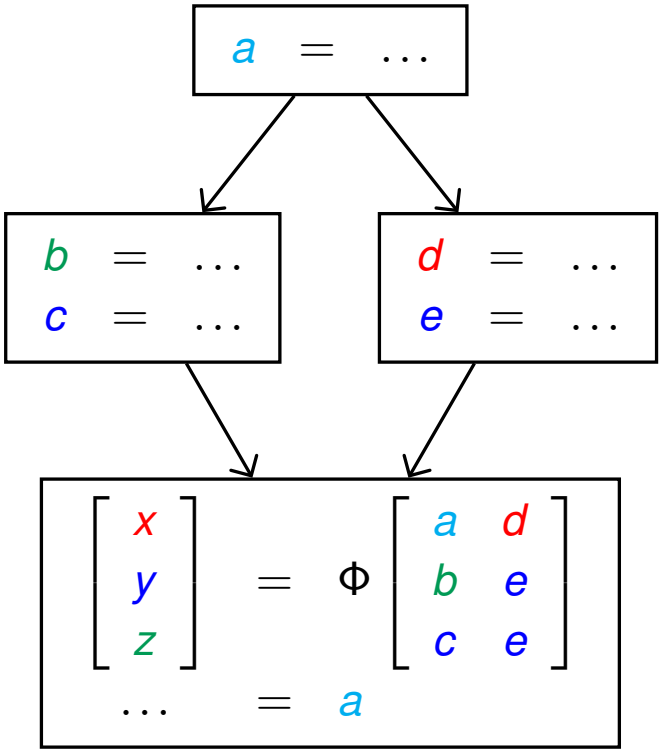
- Insert a parallel copy before each constrained operation *op*

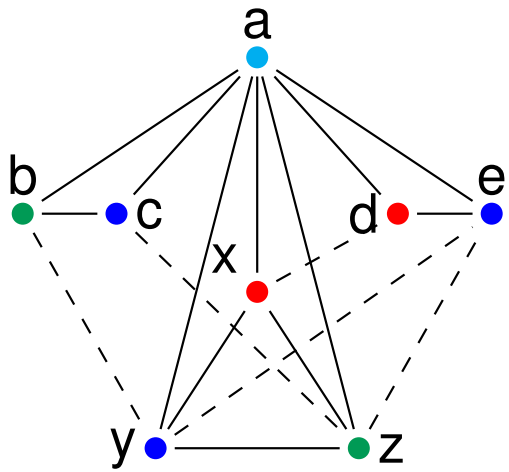
$$\begin{aligned} (a',b',c') &= (a,b,c) \\ x &= op(a',b') \\ d &= c' + x \end{aligned}$$



- This live range splitting
  - maintains *register pressure* = *chromatic number*
  - introduces affinity edges

Φ-Function Handling

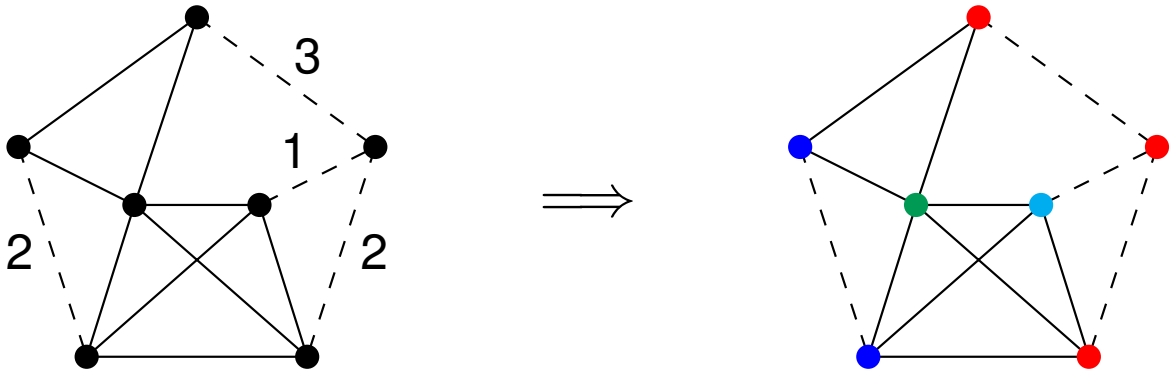




- Φs are another type of live-range splitting (SSA construction)
- Add affinity edges between each Φ result and its arguments

Outline

Instance – Solution



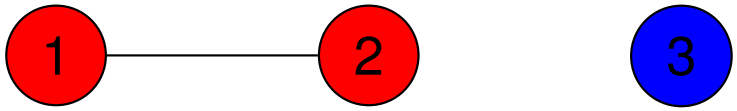
- Costs are incurred if affine nodes have different colors
- Find a correct coloring that minimizes the costs

The Basic ILP Model

The Basic ILP Model

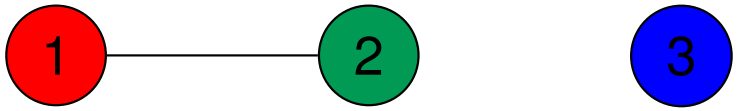
$$
\begin{array}{lll}
\text{min} & ??? & \\
\textit{where} & x_{11} + x_{12} + x_{13} = 1 & \\
 & x_{21} + x_{22} + x_{23} = 1 & \text{coloring} \\
 & x_{31} + x_{32} + x_{33} = 1 &
\end{array}
$$

The Basic ILP Model



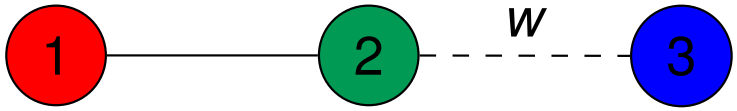
$$
\begin{array}{lll}
\text{min} & ??? & \\
\textit{where} & x_{11} + x_{12} + x_{13} = 1 & \\
 & x_{21} + x_{22} + x_{23} = 1 & \text{coloring} \\
 & x_{31} + x_{32} + x_{33} = 1 & \\
 & & \\
\lightning & x_{11} + x_{21} \leq 1 & \\
 & x_{12} + x_{22} \leq 1 & \text{interference} \\
 & x_{13} + x_{23} \leq 1 &
\end{array}
$$

The Basic ILP Model



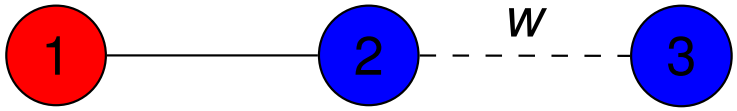
$$
\begin{array}{lll}
\text{min} & ??? & \\
\textit{where} & x_{11} + x_{12} + x_{13} = 1 & \\
 & x_{21} + x_{22} + x_{23} = 1 & \text{coloring} \\
 & x_{31} + x_{32} + x_{33} = 1 & \\
 & & \\
 & x_{11} + x_{21} \leq 1 & \\
 & x_{12} + x_{22} \leq 1 & \text{interference} \\
 & x_{13} + x_{23} \leq 1 &
\end{array}
$$

The Basic ILP Model



$$
\begin{array}{lll}
\text{min} & w \cdot y_{23} & \\
\textit{where} & x_{11} + x_{12} + x_{13} = 1 & \\
 & x_{21} + x_{22} + x_{23} = 1 & \text{coloring} \\
 & x_{31} + x_{32} + x_{33} = 1 & \\
 & & \\
 & x_{11} + x_{21} \leq 1 & \\
 & x_{12} + x_{22} \leq 1 & \text{interference} \\
 & x_{13} + x_{23} \leq 1 & \\
 & & \\
 & y_{23} \geq x_{21} - x_{31} & \\
 & y_{23} \geq x_{22} - x_{32} & \text{affinity} \\
 & y_{23} \geq x_{23} - x_{33} &
\end{array}
$$

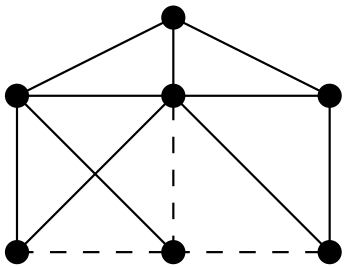
The Basic ILP Model



$$
\begin{array}{lll}
\text{min} & w \cdot y_{23} & \\
\textit{where} & x_{11} + x_{12} + x_{13} = 1 & \\
 & x_{21} + x_{22} + x_{23} = 1 & \text{coloring} \\
 & x_{31} + x_{32} + x_{33} = 1 & \\
 & & \\
 & x_{11} + x_{21} \leq 1 & \\
 & x_{12} + x_{22} \leq 1 & \text{interference} \\
 & x_{13} + x_{23} \leq 1 & \\
 & & \\
 & y_{23} \geq x_{21} - x_{31} & \\
 & y_{23} \geq x_{22} - x_{32} & \text{affinity} \\
 & y_{23} \geq x_{23} - x_{33} &
\end{array}
$$

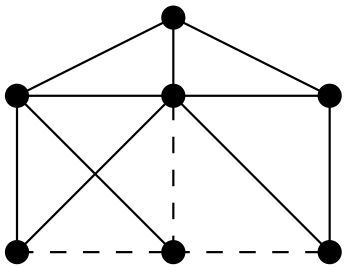
Optimizations: Preprocessing

- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



Optimizations: Preprocessing

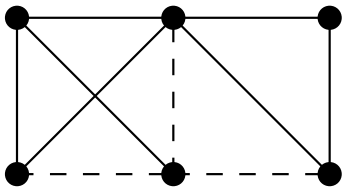
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related

Optimizations: Preprocessing

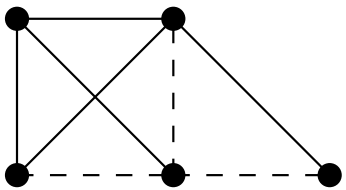
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related

Optimizations: Preprocessing

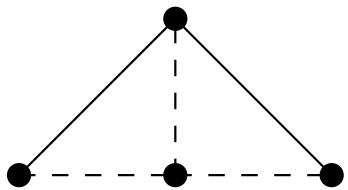
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related

Optimizations: Preprocessing

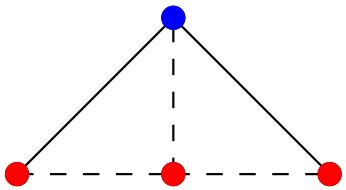
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related

Optimizations: Preprocessing

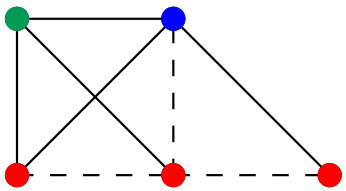
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related
- Solve the core problem

Optimizations: Preprocessing

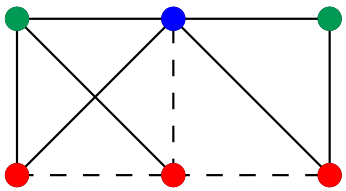
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related
- Solve the core problem
- Reinsert nodes in reverse order and color them

Optimizations: Preprocessing

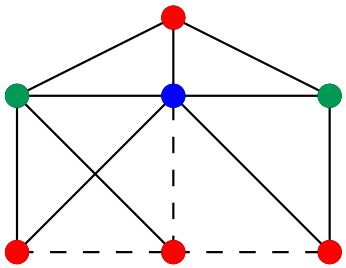
- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



- Remove maximum number of nodes not affinity-related
- Solve the core problem
- Reinsert nodes in reverse order and color them

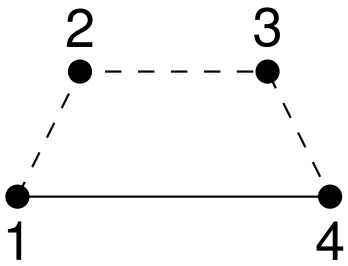
Optimizations: Preprocessing

- A node with $< k$ neighbors can always be colored
- Reduce problem to its core



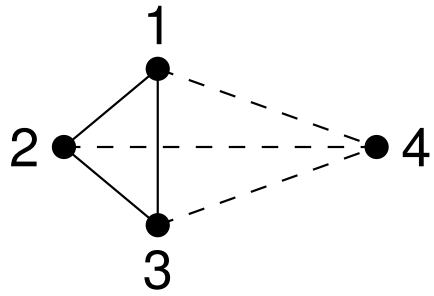
- Remove maximum number of nodes not affinity-related
- Solve the core problem
- Reinsert nodes in reverse order and color them

Optimizations: Cutting Planes



Path cut

$y_{12} + y_{23} + y_{34} \geq 1$



Clique-ray cut

$y_{14} + y_{24} + y_{34} \geq 2$

- Cutting planes limit search space
- Here: provide lower bounds on problem fragments

Outline

Optimal Coalescing Challenge

Offspring

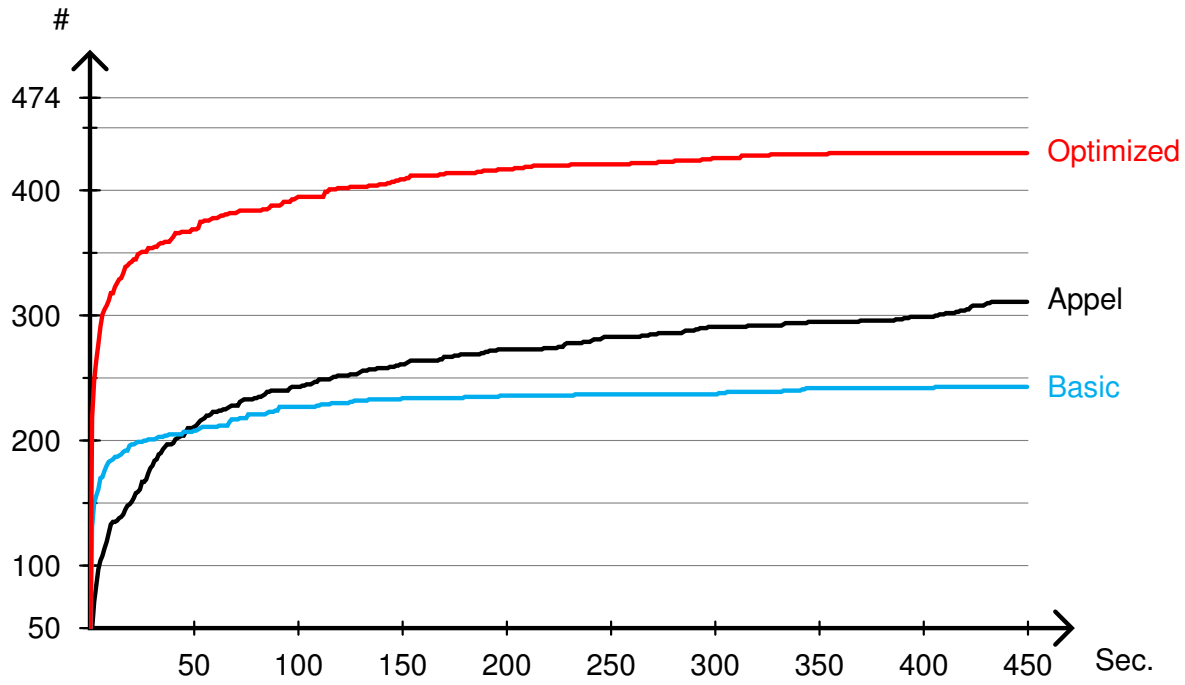
- Appel and George: Optimal Spilling for CISC Machines
- Technical report presents ILP for coalescing
- PLDI version uses optimistic coalescing

Characterization

- Set of k-colorable interference graphs
- Complete live-range splitting

$\Rightarrow$ extreme number of affinity edges

Distribution of Solution Times



- Basic model not strictly better than Appel's
- Full optimization gives good performance

Solution Quality After Time Limit (450 sec)

ILP	Appel	Optimized
Sum of all objective functions	105.3 M	2.0 M
Best lower bound given by solver	0.3 M	0.7 M

- Quality of unsolved instances (non-optimal) is better
- Good colorings are reached faster

Quality of Existing Heuristics

Algorithm	Σ Obj.	Σ off by	$\varnothing$ off by
ILP optimized	26.3 M	ε %	0 %
Iterated	66.3 M	152 %	219 %
Optimistic	40.3 M	53 %	34 %
Optimistic $_{\text{tuned}}$	37.8 M	44 %	60 %

- Supports “Optimistic better than Iterated”
- Best heuristic is 44 % off
- Large variance in quality depending on problem

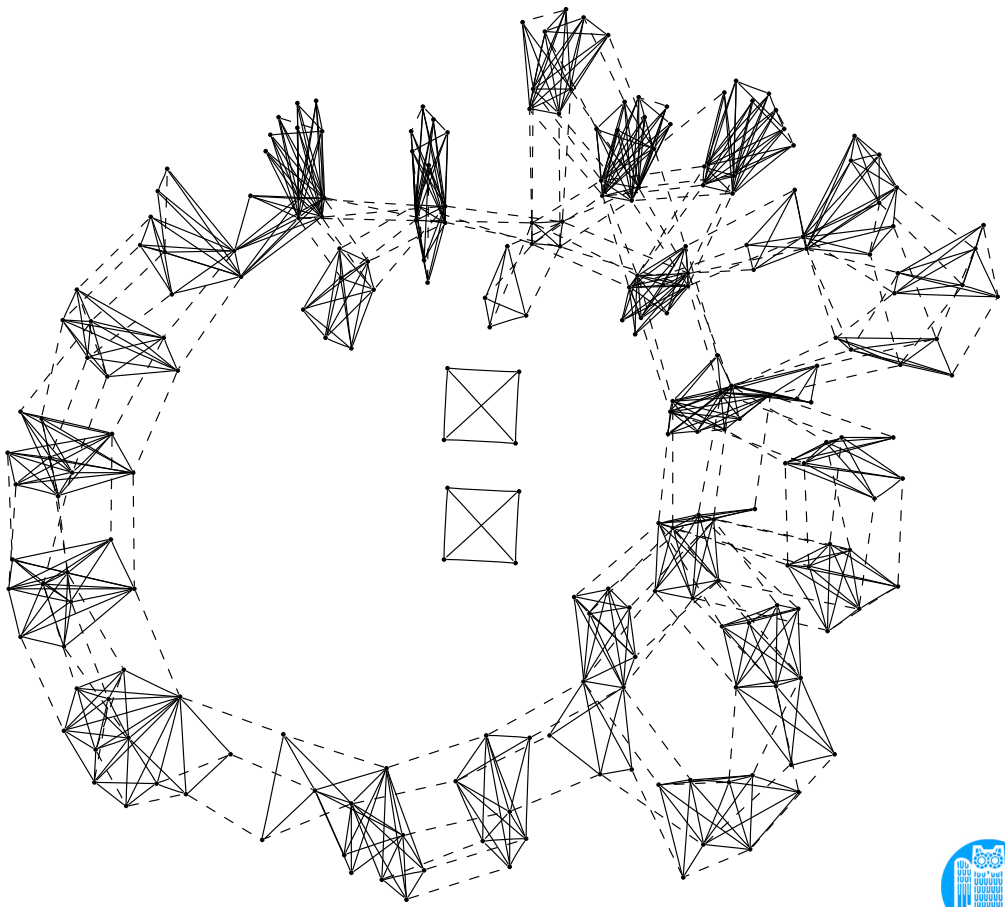
Outline

Summary

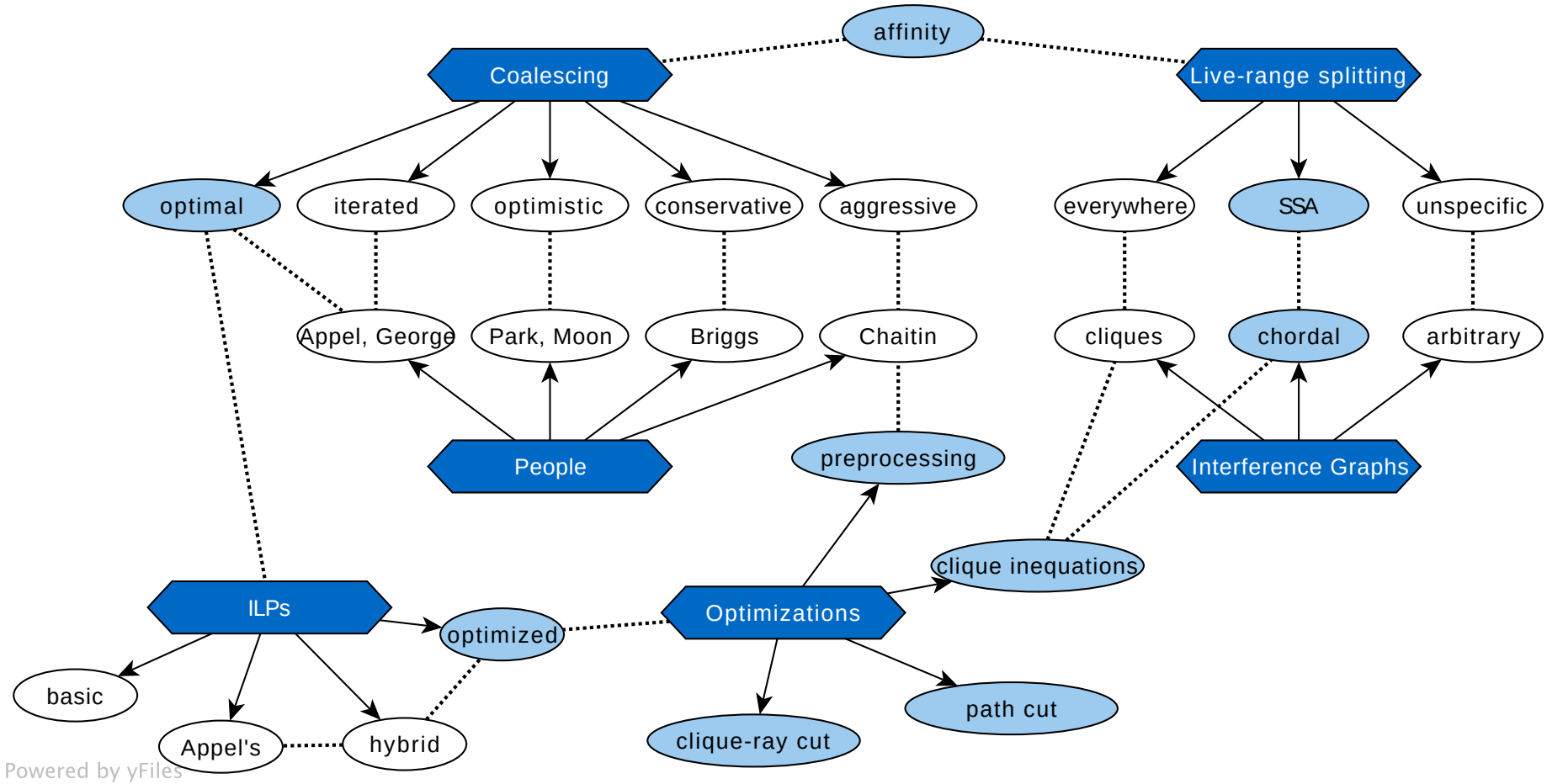
- Live-range splitting enables/preserves efficient coloring

  SSA construction $\Rightarrow$ chordal Graphs

  Constraint handling $\Rightarrow$ chordal, pre-colored Graphs
- Complexity shifted to coalescing
- Solution by optimized 0/1 LP
- First empirical assessment of heuristics



Questions?


affinity
Coalescing
Live-range splitting
optimal
iterated
optimistic
conservative
aggressive
everywhere
SSA
unspecific
Appel, George
Park, Moon
Briggs
Chaitin
cliques
chordal
arbitrary
People
preprocessing
Interference Graphs
clique inequations
ILPs
optimized
Optimizations
basic
Appel's
hybrid
clique-ray cut
path cut
Powered by yFiles

IFGs – arbitrary $\longrightarrow$ chordal $\longrightarrow$ cliques

- Chaitin: $\forall G \, \exists \, program \, P : IFG(P) = G$
- Live-range splitting enables efficient coloring

  SSA construction $\Rightarrow$ chordal Graphs

  Constraint handling $\Rightarrow$ chordal, pre-colored Graphs
- Appel (Challenge): Split everywhere $\Rightarrow$ set of cliques

  opt. live-range splitting vs. *opt. coalescing*

What about SPEC CPU 2000?

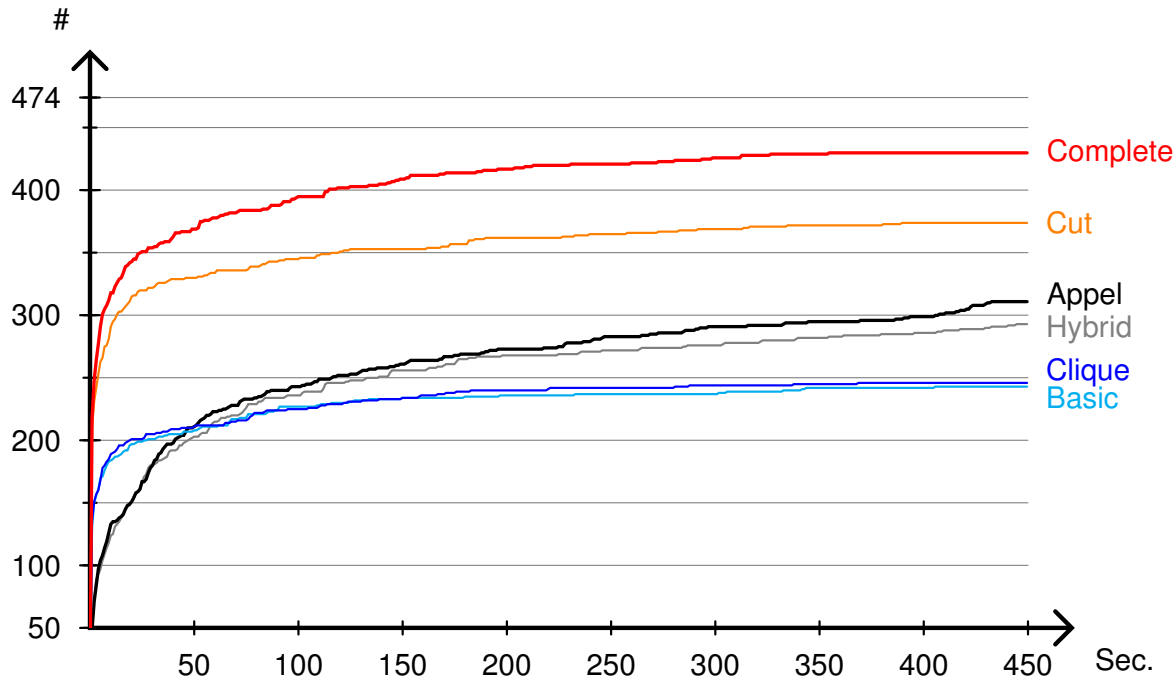
	max costs	5min	blb 5min	%5min	%blb 5min
164.gzip	3456885	97356	30935	2.82	0.89
175.vpr	17105748	218105	215758	1.28	1.26
176.gcc	221483452	3429671	2641368	1.55	1.19
181.mcf	136390	6925	4567	5.08	3.35
186.crafty	27781660	852833	390419	3.07	1.41
197.parser	22227033	678415	609249	3.05	2.74
253.perlbmk	49321969	1596567	1424011	3.24	2.89
254.gap	131547137	2908392	1930799	2.21	1.47
255.vortex	28572683	1292513	1248252	4.52	4.37
256.bzip2	7007580	239528	196840	3.42	2.81
300.twolf	162497955	2915713	1253567	1.79	0.77

- blb = best lower bound (given by ILP solver)
- Feasible. Most functions easy.

What about performance?

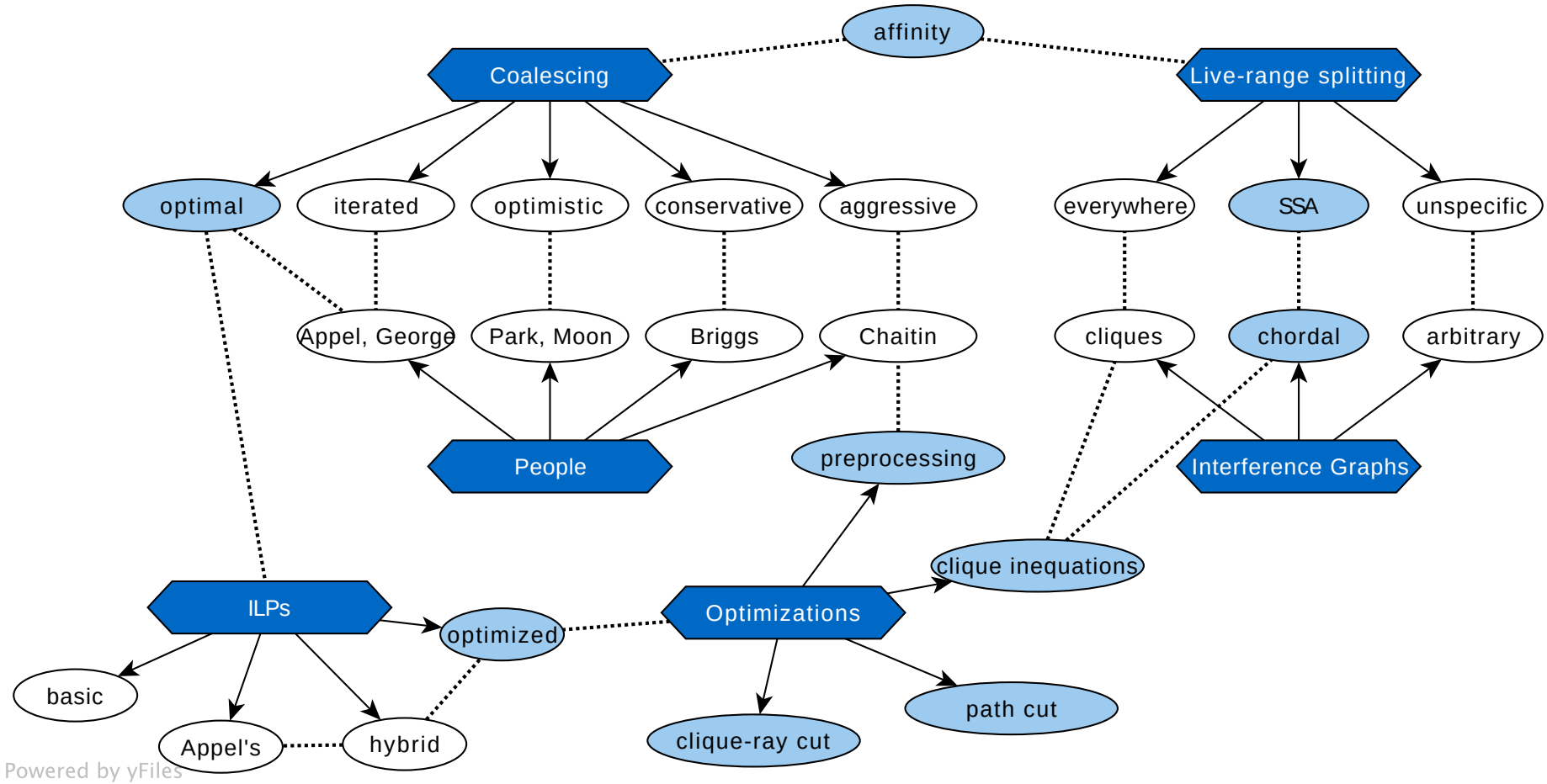
- Runtime of BZIP2 with SPEC reference input
  - Without coalescing: 314 sec
  - With a heuristic: 234 sec
  - Optimal: 229 sec
- Coalescing necessary for SSA register allocators
- Optimality less important for superscalar architectures

All Distributions of Solution Times



- Small difference between Basic and Clique
- Cut: Only the cuts presented here
- Hybrid: Combination Appels model with our cuts

Questions?


affinity
Coalescing
Live-range splitting
optimal
iterated
optimistic
conservative
aggressive
everywhere
SSA
unspecific
Appel, George
Park, Moon
Briggs
Chaitin
cliques
chordal
arbitrary
People
preprocessing
Interference Graphs
clique inequations
ILPs
optimized
Optimizations
basic
Appel's
hybrid
clique-ray cut
path cut
Powered by yFiles

Summary

- Live-range splitting enables/preserves efficient coloring

  SSA construction $\Rightarrow$ chordal Graphs

  Constraint handling $\Rightarrow$ chordal, pre-colored Graphs
- Complexity shifted to coalescing
- Solution by optimized 0/1 LP
- First empirical assessment of heuristics