

AHC001

komori3

# 概要

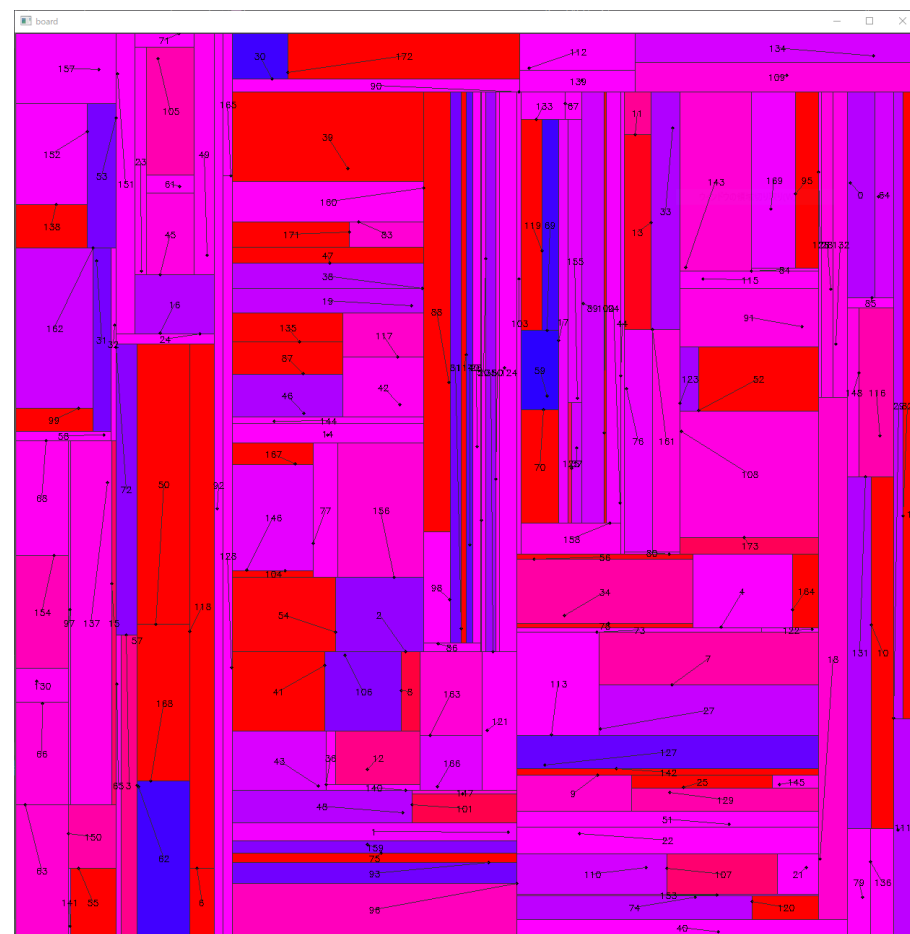
- 初期解候補生成
  - 縦割りと横割りを繰り返して領域を分割する
  - ランダム性を持たせて候補をたくさん生成
- 初期解選択
  - ちょっと山登りしたあとのスコアで評価
- 焼きなまし
- 後処理

# 初期解候補生成

- 縦割りと横割りを繰り返して領域を分割する

# 初期解候補生成

- 縦割りと横割りを繰り返して領域を分割する
  - こんな感じ



# 初期解候補生成

- 縦割りと横割りを繰り返して領域を分割する
  - こんな感じ
  - スコア見積もりのためにフィット



# 初期解候補生成

- 縦割りと横割りを繰り返して領域を分割する
  - こんな感じ
  - スコア見積もりのためにフィット
- たくさん生成する

```
1168 vector<pair<double, SplitSolver::StatePtr>> svec;  
1169 {  
1170     double best = -1.0;  
1171     int loop = 0;  
1172     while (timer.elapsedMs() < time_split) {  
1173         SplitSolver::StatePtr stmp = make_shared<SplitSolver::State>(N, points);  
1174         stmp->recursive_random_split(rnd, stmp->root, 10, 0.95 + rnd.next_double() * 0.04);  
1175         double score = stmp->calc_score();  
1176         svec.emplace_back(score, stmp);  
1177         if (best < score) {  
1178             best = score;  
1179             dump(timer.elapsedMs(), best);  
1180         }  
1181         loop++;  
1182     }  
1183     dump(best, loop);  
1184 }  
1185 dump(svec.size());  
1186 sort(svec.begin(), svec.end(), [](const auto& a, const auto& b) { return a.first < b.first; });
```

1.5 sec


ランダムで悪あがき



# 初期解選択

- ちょっと山登りしたあとのスコアで評価
  - 初期解候補をスコアの高いものから取り出す
  - 100msec 山登り
  - スコア改善したなら初期解 (best\_state) を更新
  - 2.5 sec まで回す

```
1188     /* climbing */
1189     double best_score = -1.0;
1190     LocalSearchSolver::State best_state;
1191     while (!svec.empty() && timer.elapsedMs() < time_climbing) {
1192         const auto [score, sstate] = svec.back(); svec.pop_back();
1193         LocalSearchSolver::State astate = sstate->cvt();
1194         LocalSearchSolver::Solver solver(astate);
1195         solver.climbing(100);
1196         if (best_score < solver.best_state.score) {
1197             best_score = solver.best_state.score;
1198             best_state = solver.best_state;
1199             ll::lls = (ll)round(best_state.score * 1e9 / best_state.N);
1200             dump(score, best_state.score, N, lls);
1201         }
1202     }
1203     dump(best_state.score);
1204
```



# 山登り遷移

- ForceDeform(id, dir, inflate)
  - 長方形 id を dir 方向に inflate だけ膨張させる
  - 他の長方形と重なったら他の長方形を収縮させる

```
432
433     void climbing(double process_ms){
434         double tend = timer.elapsedMs() + process_ms;
435         while (timer.elapsedMs() < tend){
436             for (int rid = 0; rid < best_state.N; rid++){
437                 for (int d = 0; d < 4; d++){
438                     ForceDeform trans(rid, d, 1);
439                     auto ok = best_state.is_acceptable(trans);
440                     if (!ok) continue;
441                     if (trans.sdiff > 0){
442                         best_state.accept(trans);
443                     }
444                 }
445             }
446         }
447     }
448 }
```

```
1188     /* climbing */
1189     double best_score = -1.0;
1190     LocalSearchSolver::State best_state;
1191     while (!svec.empty() && timer.elapsedMs() < time_climbing){
1192         const auto [score, sstate] = svec.back(); svec.pop_back();
1193         LocalSearchSolver::State astate = sstate->cvt();
1194         LocalSearchSolver::Solver solver(astate);
1195         solver.climbing(100);
1196         if (best_score < solver.best_state.score){
1197             best_score = solver.best_state.score;
1198             best_state = solver.best_state;
1199             ll_l1s = (ll)round(best_state.score * 1e9 / best_state.N);
1200             dump(score, best_state.score, N, ll_s);
1201         }
1202     }
1203     dump(best_state.score);
1204 }
```

2.5 sec



# 焼きなまし

- 遷移: Deform(id, dir, inflate)
  - 長方形 id を dir 方向に inflate だけ膨張させる (inflate 負なら収縮)
  - 他の長方形と重なる遷移は許さない

```
1205  /* annealing */
1206  LocalSearchSolver::Solver annealer(best_state);
1207  annealer.annealing(time_annealing - timer.elapsedMs(), 0.03, 0.0);
1208  dump(annealer.best_state.score, timer.elapsedMs());
1209  annealer.climbing(400);
1210  dump(annealer.best_state.score, timer.elapsedMs());
1211  dump((ll)round(annealer.best_state.score * 1e9 / N));
1212  annealer.best_state.output(out);
1213
```

~4.5 sec

# 焼きなまし

- 遷移: Deform(id, dir, inflate)
  - 長方形 id を dir 方向に inflate だけ膨張させる (inflate 負なら収縮)
  - 他の長方形と重なる遷移は許さない
- パラメータ
  - 初期温度 0.03
  - Inflate 幅は 500 → 10 で線形に減少

```
1205  /* annealing */
1206  LocalSearchSolver::Solver annealer(best_state);
1207  annealer.annealing(time_annealing - timer.elapsedMs(), 0.03, 0.0);
1208  dump(annealer.best_state.score, timer.elapsedMs());
1209  annealer.climbing(400);
1210  dump(annealer.best_state.score, timer.elapsedMs());
1211  dump((11)round(annealer.best_state.score * 1e9 / N));
1212  annealer.best_state.output(out);
1213
```

~4.5 sec

```
394  auto get_temp = [](double stemp, double etemp, double loop, double num_loop) {
395      return etemp + (stemp - etemp) * (num_loop - loop) / num_loop;
396  };
397  auto get_width = [](double tbegin, double tend, double t) {
398      return 10.0 + (500.0 - 10.0) * (tend - t) / (tend - tbegin);
399  };
```

# 焼きなまし

- 遷移: Deform(id, dir, inflate)
  - 長方形 id を dir 方向に inflate だけ膨張させる (inflate 負なら収縮)
  - 他の長方形と重なる遷移は許さない
- パラメータ
  - 初期温度 0.03
  - Inflate 幅は 500 → 10 で線形に減少

```
1205  /* annealing */
1206  LocalSearchSolver::Solver annealer(best_state);
1207  annealer.annealing(time_annealing - timer.elapsedMs(), 0.03, 0.0);
1208  dump(annealer.best_state.score, timer.elapsedMs());
1209  annealer.climbing(400);
1210  dump(annealer.best_state.score, timer.elapsedMs());
1211  dump((ll)round(annealer.best_state.score * 1e9 / N));
1212  annealer.best_state.output(out);
1213
```

~4.5 sec

```
394  auto get_temp = [](double stemp, double etemp, double loop, double num_loop) {
395      return etemp + (stemp - etemp) * (num_loop - loop) / num_loop;
396  };
397  auto get_width = [](double tbegin, double tend, double t) {
398      return 10.0 + (500.0 - 10.0) * (tend - t) / (tend - tbegin);
399  };
```

- 最後にダメ押しの山登り 0.4sec

# 焼きなまし

- 遷移: Deform(id, dir, inflate)
  - 長方形 id を dir 方向に inflate だけ膨張させる (inflate 負なら収縮)
  - 他の長方形と重なる遷移は許さない
- パラメータ
  - 初期温度 0.03
  - Inflate 幅は 500 → 10 で線形に減少

```
1205  /* annealing */
1206  LocalSearchSolver::Solver annealer(best_state);
1207  annealer.annealing(time_annealing - timer.elapsedMs(), 0.03, 0.0);
1208  dump(annealer.best_state.score, timer.elapsedMs());
1209  annealer.climbing(400);
1210  dump(annealer.best_state.score, timer.elapsedMs());
1211  dump((ll)round(annealer.best_state.score * 1e9 / N));
1212  annealer.best_state.output(out);
1213
```

~4.5 sec

```
394  auto get_temp = [](double stemp, double etemp, double loop, double num_loop) {
395      return etemp + (stemp - etemp) * (num_loop - loop) / num_loop;
396  };
397  auto get_width = [](double tbegin, double tend, double t) {
398      return 10.0 + (500.0 - 10.0) * (tend - t) / (tend - tbegin);
399  };
```

- 最後にダメ押しの山登り 0.4sec

