May 19, 2018 at 02:30

1. Intro. This little program outputs clauses that are satisfiable if and only if the graph g can be c-colored with kernels, given g and c.

(It generalizes SAT-PIGEONS, which is the case where $g = K_m$ and $c = n$.)

Suppose the graph has m edges and n vertices. Then there are nc variables $v.k$, meaning that vertex v gets color k. And there are n clauses of size c (to ensure that each vertex gets at least one color), plus mc clauses of size 2 (to ensure that adjacent vertices don't share a color). Plus nc clauses for each extended neighborhood.

```
#include <stdio.h>
#include <stdlib.h>
#include "gb_graph.h"
#include "gb_save.h"
  int c;

  main(int argc, char *argv[])
  {
    register int i, j, k;
    register Arc *a;
    register Graph *g;
    register Vertex *v;

    ⟨ Process the command line 2 ⟩;
    ⟨ Generate the positive clauses 3 ⟩;
    ⟨ Generate the negative clauses 4 ⟩;
    ⟨ Generate the kernel clauses 5 ⟩;
  }
```

2. ⟨ Process the command line 2 ⟩ ≡

```
  if (argc ≠ 3 ∨ sscanf(argv[2], "%d", &c) ≠ 1) {
    fprintf(stderr, "Usage:␣%s␣foo.gb␣c\n", argv[0]);
    exit(−1);
  }
  g = restore_graph(argv[1]);
  if (¬g) {
    fprintf(stderr, "I␣couldn't␣reconstruct␣graph␣%s!\n", argv[1]);
    exit(−2);
  }
  if (c ≤ 0) {
    fprintf(stderr, "c␣must␣be␣positive!\n");
    exit(−3);
  }
  printf("~␣sat-color-kernel␣%s␣%d\n", argv[1], c);
```

This code is used in section 1.

3. ⟨ Generate the positive clauses 3 ⟩ ≡

```
  for (v = g→vertices; v < g→vertices + g→n; v++) {
    for (k = 1; k ≤ c; k++) printf("␣%s.%d", v→name, k);
    printf("\n");
  }
```

This code is used in section 1.

4. ⟨ Generate the negative clauses 4 ⟩ ≡

```
for (k = 1; k ≤ c; k++)
  for (v = g→vertices; v < g→vertices + g→n; v++)
    for (a = v→arcs; a; a = a→next)
      if (a→tip > v) printf("~%s.%d␣~%s.%d\n", v→name, k, a→tip→name, k);
```

This code is used in section 1.

5. ⟨ Generate the kernel clauses 5 ⟩ ≡

```
for (k = 1; k ≤ c; k++)
  for (v = g→vertices; v < g→vertices + g→n; v++) {
    printf("%s.%d", v→name, k);
    for (a = v→arcs; a; a = a→next) printf("␣%s.%d", a→tip→name, k);
    printf("\n");
  }
```

This code is used in section 1.

6. Index.

⟨ Generate the kernel clauses 5 ⟩ Used in section 1.
⟨ Generate the negative clauses 4 ⟩ Used in section 1.
⟨ Generate the positive clauses 3 ⟩ Used in section 1.
⟨ Process the command line 2 ⟩ Used in section 1.

SAT-COLOR-KERNEL