§1 SAT-COLOR-SNARK1 INTRO 1

May 19, 2018 at 02:31

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

```
#include <stdio.h>
#include <stdlib.h>
#include "gb_graph.h"
#include "gb_save.h"
  int c;
  int n;
               /* order of flower snark line graph (a command-line parameter) */
  char buf[20];
  main(\mathbf{int} \ argc, \mathbf{char} * argv[])
  {
     register int i, j, k;
     register Arc *a;
     register Graph *g;
     register Vertex *v;
     \langle \text{Process the command line } 2^* \rangle;
     (Generate the positive clauses 3);
     \langle Generate the negative clauses 4\rangle;
  }
2* \langle Process the command line 2^*\rangle \equiv
  if (argc \neq 2 \lor sscanf(argv[1], "%d", &n) \neq 1) {
     fprintf(stderr, "Usage: \_\%s \_n \n", argv[0]);
     exit(-1);
  sprintf(buf, "fsnarkline%d.gb", n);
  g = restore\_graph(buf);
  if (\neg g) {
     fprintf(stderr, "I_{\square}couldn't_{\square}reconstruct_{\square}graph_{\square}%s! \n", buf);
     exit(-2);
  }
  c = 3;
  printf("\"alpha", n);
  printf("b1.1\n");
                            /* start with three unary clauses to break symmetry */
  printf("c1.2\n");
  printf("d1.3\n");
This code is used in section 1*.
3. \langle Generate the positive clauses 3\rangle \equiv
  for (v = g \rightarrow vertices; v < g \rightarrow vertices + g \rightarrow n; v \leftrightarrow) {
     for (k = 1; k \le c; k++) printf("\_%s.%d", v \rightarrow name, k);
     printf("\n");
This code is used in section 1*.
```

2 INTRO SAT-COLOR-SNARK1 §4

```
4. \langle Generate the negative clauses 4 \rangle \equiv for (k=1; k \leq c; k++) for (v=g \neg vertices; v < g \neg vertices + g \neg n; v++) for (a=v \neg arcs; a; a=a \neg next) if (a \neg tip > v) printf ("~%s.%d\n", v \neg name, k, a \neg tip \neg name, k); This code is used in section 1*.
```

 $\S 5$ Sat-color-snark1 index 3

5* Index.

The following sections were changed by the change file: 1, 2, 5.

a: <u>1</u>* **Arc**: 1* arcs: 4.argc: 1,* 2.* argv: 1,* 2.* buf: 1,* 2.* c: 1.* exit: 2*fprintf: 2*g: <u>1</u>* Graph: 1.* i: $\underline{1}$ * j: $\underline{\underline{1}}^*$ k: $\underline{\underline{1}}^*$ $main: \underline{1}^*$ $n: \underline{1}^*$ name: 3, 4.next: 4.printf: 2, 3, 4.restore_graph: 2* sprintf: 2*
sscanf: 2* stderr: 2*tip: 4.v: $\underline{1}$ * Vertex: 1.* vertices: 3, 4.

4 NAMES OF THE SECTIONS

SAT-COLOR-SNARK1

 $\begin{array}{ll} \langle \, \text{Generate the negative clauses} \,\, 4 \, \rangle & \text{Used in section } 1^*. \\ \langle \, \text{Generate the positive clauses} \,\, 3 \, \rangle & \text{Used in section } 1^*. \\ \langle \, \text{Process the command line } \, 2^* \, \rangle & \text{Used in section } 1^*. \end{array}$

SAT-COLOR-SNARK1

	Section	Page
Intro	 1	1
Index	5	3