Experimental Descriptive Complexity

Charles Jordan

Joint work with Marco Carmosino and Neil Immerman.
Festschrift for Dexter Kozen, LNCS 7230, pp. 24–34, 2012.

DE: `http://www.cs.umass.edu/~immerman/de`

May 18, 2012

Our Environment

DescriptiveEnvironment (DE)

DE is a *playground* for (descriptive) complexity theorists.

- Implements objects and tools of descriptive complexity.
- Similar to Mathematica in spirit.
- Helps us use computers to attack big problems.

Research finding counter-examples, reductions, examining complicated maps between problems, visualization, ...

Teaching descriptive complexity, finite model theory, logic, complexity theory, ...

Development *rapid programming*, e.g., easily using SAT solvers as general-purpose solvers in a rigorous way.

Outline

Relational Structures in DE

Examples in DE

```
string is new vocabulary{S:1}.
graph is new vocabulary{E:2, s, t}.
```

Relational Structures in DE

Examples in DE

```
string is new vocabulary{S:1}.
graph is new vocabulary{E:2, s, t}.

primes100 is new structure{string,100, S:1 is (1<x1 &
    \A x, y.(x<=x1 & y<=x) :  ((x*y=x1)->(x=1|y=1)))}.
```

Relational Structures in DE

Examples in DE

```
string is new vocabulary{S:1}.
graph is new vocabulary{E:2, s, t}.

primes100 is new structure{string,100, S:1 is (1<x1 &
    \A x, y.(x<=x1 & y<=x) :  ((x*y=x1)->(x=1|y=1)))}.
line10 is new structure{graph, 10, E:2 is x2=x1+1,
    s is 0, t is 9}.
```

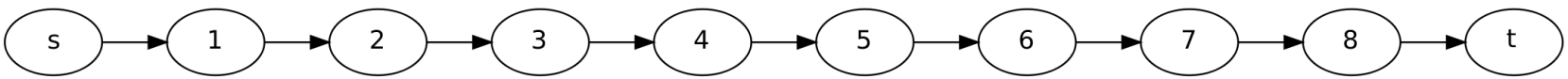
Relational Structures in DE

Examples in DE

```
string is new vocabulary{S:1}.
graph is new vocabulary{E:2, s, t}.

primes100 is new structure{string,100, S:1 is (1<x1 &
    \A x, y.(x<=x1 & y<=x) :  ((x*y=x1)->(x=1|y=1)))}.
line10 is new structure{graph, 10, E:2 is x2=x1+1,
    s is 0, t is 9}.

primes100.S.
:{(2), (3), (5), (7), (11), (13), (17), (19), (23),
(29), (31), (37), (41), (43), (47), (53), (59), (61),
(67), (71), (73), (79), (83), (89), (97)}
draw(line10).
```



Queries

Query (Computation)

Let vocabulary $\tau := \{R_1^{a_1}, \ldots, R_s^{a_s}, c_1, \ldots, c_t\}$.

A query $f : STRUC(\sigma) \to STRUC(\tau)$ is an $(s+t+2)$-tuple,

$$\langle k, \varphi_0, \varphi_1, \ldots, \varphi_s, \psi_1, \ldots, \psi_t \rangle .$$

Let $A \in STRUC(\sigma)$. Then, $f(A)$ is defined as follows:

1. Universe U is $\{\langle x_1, \ldots, x_k \rangle \mid x_i \in U^A, (A, \langle x_1, \ldots, x_k \rangle) \models \varphi_0\rangle$.
2. $R_i^{f(A)}$ is

$$U^{a_i} \cap \{\langle \langle x_1^1, \ldots, x_k^1 \rangle, \ldots, \langle x_1^{a_i}, \ldots, x_k^{a_i} \rangle \rangle \mid (A, \ldots) \models \varphi_i\} .$$

3. c_j is the unique $\langle x_1, \ldots, x_k \rangle$ such that $(A, \langle x_1, \ldots, x_k \rangle) \models \psi_j$.

Queries in DE

Examples in DE

```
Reach is new bquery{graph, TC[x,y:E(x,y)](s,t)}.
```

Queries in DE

Examples in DE

```
Reach is new bquery{graph, TC[x,y:E(x,y)](s,t)}.
Reach(line10).
:\t

thrcolor is new bquery{graph, \E R:1 :  \E G:1 :  \E
B:1 :  \A x, y :  ((R(x)|G(x)|B(x)) &
(E(x,y)->~((R(x)&R(y))|(G(x)&G(y))|(B(x)&B(y)))))}.
thrcolor(line10).
:\t
```

Queries in DE

Examples in DE

```
Reach is new bquery{graph, TC[x,y:E(x,y)](s,t)}.
Reach(line10).
:\t

thrcolor is new bquery{graph, \E R:1 :  \E G:1 :  \E
B:1 :  \A x, y :  ((R(x)|G(x)|B(x)) &
(E(x,y)->~((R(x)&R(y))|(G(x)&G(y))|(B(x)&B(y)))))}.
thrcolor(line10).
:\t

I is new query{graph,string,2,\t,S:1 is E(x1,x2)}.
set100 is I(line10).
set100.S.
```

Queries in DE

Examples in DE

```
Reach is new bquery{graph, TC[x,y:E(x,y)](s,t)}.
Reach(line10).
:\t

thrcolor is new bquery{graph, \E R:1 :  \E G:1 :  \E
B:1 :  \A x, y :  ((R(x)|G(x)|B(x)) &
(E(x,y)->~((R(x)&R(y))|(G(x)&G(y))|(B(x)&B(y)))))}.
thrcolor(line10).
:\t

I is new query{graph,string,2,\t,S:1 is E(x1,x2)}.
set100 is I(line10).
set100.S.
:{(1), (12), (23), (34), (45), (56), (67), (78), (89)}
```

Reductions in DE

A *reduction* r from P to P' is a query such that

$$A \in P \iff r(A) \in P'.$$

Examples in DE

```
thrcoltosat is new reduction{graph, sat, 3, x1<=3,
P:2 is x1=3&x4=0&x2=0&x3=x6&x5<3,
N:2 is x4=0 & E(x2,x3) & (x1=x5 & (x6=x2 | x6=x3)) &
       x1<3}.
scline is thrcoltosat(line10).
minisat(scline).
:\t
threecolorwithsat(line10).
:\t
```

Finding Reductions

Examples in DE

```
a1 is new bquery{string, \forall x:S(x)}.
a0 is new bquery{string, \forall x:~S(x)}.
redfind(a1,a0).
:  A0:=new reduction{string,string,1,\t,
                     S:1 is (~S(x1))}.
```

Finding Reductions

Examples in DE

```
a1 is new bquery{string, \forall x:S(x)}.
a0 is new bquery{string, \forall x:~S(x)}.
redfind(a1,a0).
:  A0:=new reduction{string,string,1,\t,
                     S:1 is (~S(x1))}.

rs is new vocabulary{R:1, S:1}.
gr is new vocabulary{E:2}.
eras is new bquery{rs, \exists x:R(x)&\forall y:S(y)}.
axey is new bquery{gr, \forall x:\exists y:E(x,y)}.
redfind(eras,axey).
:  A0:=new reduction{rs,gr,1,\t,E:2 is (S(x1)&R(x2))}.
```

Outline

What does DE do?

- Creating vocabularies and structures.
- Defining queries and reductions in SO∃(TC).
- Calling Mace4[1] to get a structure that satisfies a desired FO property.
- Using `minisat()` and `zchaff()` to use built-in SAT solvers MiniSat and zChaff to evaluate Boolean satisfiability.
- Loading structures (DIMACS-format graphs, etc.).
- Saving and drawing structures.

[1] A first-order model-finding program.

What will DE do?

1. Adding additional operators (IFP, etc.).
2. Try to find reductions (in progress).
3. Play Ehrenfeucht-Fraïssé games.
4. Visualizing queries, TC, etc.
5. Example-driven query construction.
6. Other UIs.
7. Possibly interact with automatic synthesis of fast programs.
8. New functionality with plug-ins.

Outline

Finding Reductions I

ReductionFinder

Crouch, Immerman and Moss (2010) developed a reduction finding tool. We are integrating an improved version into DE.

Basic idea (trying $A \leqslant B$):

1. Take a random G_0 of small size. Ask the SAT solver for a simple reduction f_0 s.t. $G_0 \in A \iff f(G_0) \in B$.
2. Ask the SAT solver for a small G_1 that's a counter-example to f (i.e., $G_1 \in A \oplus f(G_1) \in B$).
3. Repeat: ask for a new reduction on the examples so far, then ask for a new counter-example.

Finding Reductions II

NL=coNL

NI's original proof of NL=coNL constructed an 8-ary, quantifier-free projection from $\overline{\text{REACH}}$ to REACH.

Are computers fast enough to put this reduction in reach of automatic discovery? ReductionFinder is too slow; possible to do better.

Question

What else could be found?

Property Testing

Recent Proofs by CJ

There are untestable properties in $[\forall\exists\forall, (0,1)]$ in one model of testability.
To extend to other models,

- are large models of a particular FO formula probably far from models of another?
- I'd like to check (large) examples – adding distance computation to DE would help.

The distances are probably non-trivial to compute.

Playing EF Games

Plans

We plan to automate EF games:

1. Playing the computer.
2. The computer plays itself.
3. Analysis of strategies.

Uses

Can DE help with complicated games, like those for FO(rank,LFP)?

Outline

Programming with Reductions I

Basic Problem

Many times, we have some problem we want to solve. So, we write a program to solve it.
Problems:

- Error-prone and time-consuming.
- Hard to produce a program that is really optimized.

Programming with Reductions I

Basic Problem

Many times, we have some problem we want to solve. So, we write a program to solve it.
Problems:

- Error-prone and time-consuming.
- Hard to produce a program that is really optimized.

Programming with Reductions

Write a reduction to (e.g.,) SAT and use a good SAT solver.

- Usually, these reductions are easy.
- Computation is dominated by SAT solver.
- Leverage research effort on SAT solvers.
- Easy and fast development of *decently*-optimized programs.

Programming with Reductions II

Examples

- Kugele. Efficient Solving of Combinatorial Problems using SAT-Solvers. Diplomarbeit in Informatik, TU München, 2006.
- Ramani, Aloul, Markov and Sakallah. Breaking Instance-Independent Symmetries in Exact Graph Coloring. J. Artif. Intell. Res. 26:289–322, 2006.
- Mitchell and Ternovska. Knowledge Representation, Search Problems and Model Expansion. Knowing, Reasoning and Acting, 347–362, 2011.
- Joao Marques-Silva. Practical applications of Boolean satisfiability. Proc. WODES 2008, 74–80, 2008.

Why use descriptive complexity?

Programming with Reductions II

Examples

- Kugele. Efficient Solving of Combinatorial Problems using SAT-Solvers. Diplomarbeit in Informatik, TU München, 2006.
- Ramani, Aloul, Markov and Sakallah. Breaking Instance-Independent Symmetries in Exact Graph Coloring. J. Artif. Intell. Res. 26:289–322, 2006.
- Mitchell and Ternovska. Knowledge Representation, Search Problems and Model Expansion. Knowing, Reasoning and Acting, 347–362, 2011.
- Joao Marques-Silva. Practical applications of Boolean satisfiability. Proc. WODES 2008, 74–80, 2008.

Why use descriptive complexity?

- A rigorous way to do what is currently done adhoc.
- *Nobody* wants to write Turing machine transition tables all day.

Outline

Summary

We implemented DescriptiveEnvironment (DE), an interactive playground for descriptive complexity.

- Freely available[2] under an ISC license.
- Usable now (we use it), plans for improvement.
- We encourage anyone to use it, please give us suggestions, improvements, etc.

[2] `http://www.cs.umass.edu/~immerman/de`

Thank you!