Paul is really an amazing resource to have around! See exchange of messages below!

> From kautz@cs.washington.edu Sat Jun 11 19:17:06 2005
> Date: Sat, 11 Jun 2005 19:17:06 -0700
> From: Henry Kautz <kautz@cs.washington.edu>
> To: clu@tcs.inf.tu-dresden.de
> CC: beame@cs.washington.edu
> Subject: Re: Propositional Succinctness
>
> That's an interesting question. I don't know the answer. I'm cc'ing Paul Beame, he might know. Paul, the question is the following:
>
> Can adding variables exponentially shrink a theory in PL? That is:

> Let F1, F2, ... be a family of formulas in propositional logic.
> Let min(F1), min(F2), ... be the family of smallest equivalent formulas.
> Can there be a family of formulas G1, G2, ...
> such that:
> 1. The variables in Gi are a superset of those in Fi;
> 2. The Gi are exponentially smaller than the Fi: |Gi| <= log |Fi|
> 3. Gi and Fi are equivalent modulo the variables not in Fi;
>    that is, every model of Fi can be extended to a model of Gi,
>    and every model of Gi satisfies Fi.
>
> This is, of course, true for restricted kinds of propositional logic,
> such as clausal form: but what about for PL in general?
>
>>
>>> Date: Sat, 11 Jun 2005 15:44:03 +0200 (CEST)
>> From: Carsten Lutz <clu@tcs.inf.tu-dresden.de>
>>>
>> Dear Henry Kautz,
>>> I just read with great interest your IJCAI-95 paper on the comparative linguistics of knowledge representation. I did this since I want to find out what is known about a particular succinctness question for propositional logic (which is not really my proper field). Since my question is very close to what is discussed in your paper, I suspect that you may have some ideas or pointers concerning my problem.
>>>
>>> My problem is easily stated using the terminology of your paper: does adding extra variables to full PL (with only the three standard Boolean operators) increase its representational succinctness (in the same sense as considered for Horn formulas in Section 5 of your paper)? Or equivalently: is the existential prefix fragment of QBF representationally more succinct than PL?
> I suspect that the answer is positive, i.e., PL becomes exponentially more
> succinct when allowing additional variables. Also I have the feeling that a
> constructive proof (i.e. the exhibition of a concrete family of formulas
> that can be represented shortly by PL+Var but not by PL) cannot be expected
> as this would produce a most-wanted example for a Boolean function without
> a short PL representation.
> 
> But perhaps there is some non-constructive argument? Or a conditional
> result of the form "PL+Var is exponentially more succinct than PL
> unless P=NP/the PH collapses/etc."?
> 
> I apologize if this should be a stupid question (as I said, this is not
> my proper field). In any case, I would be very happy if you could provide
> me with any hints and/or literature pointers concerning this problem.
> 
> Best regards and many thanks in advance,
> 
> Carsten

> From beame@cs.washington.edu Sat Jun 11 19:53:42 2005
> Date: Sat, 11 Jun 2005 19:53:41 -0700
> From: beame@cs.washington.edu
> To: clu@tcs.inf.tu-dresden.de, kautz@cs.washington.edu
> Subject: Re: Propositional Succinctness
> Cc: beame@cs.washington.edu
> 
> The question is exactly the question of circuit size versus formula size
> to represent Boolean functions. (The new variables represent the values
> of the gates of the circuits.)
> 
> If the connectives are AND OR and NOT then there is a quadratic gap between
> the two for the parity function on n bits (is there an odd # of 1's in the
> input?) - circuit can compute this in linear size but any formula using AND
> OR and NOT takes n^2 size.
> For an arbitrary set of connectives the best known difference is not
> so large. For example almost all functions have size roughly (2^n)/n for
> circuits vs size (2^n)/log n for formulas over any set of connectives which
> means that there is a difference.
> 
> No exponential gap is known but it is conjectured. If the gap were only
> polynomial then it would follow that every polynomial time computable problem
> would have a superfast parallel algorithm having running time O(log n).
> 
> Paul