Report on Promises in Satisfaction Problems by Kristina Asimi

This thesis is based on the study of certain promise problems. In a promise problem, the input is guaranteed to satisfy some properties. Thus, an algorithm for a promise problem can answer any nonsense for inputs which fail to have this property (one talks of the promise being violated). The thesis focusses on problems of the following form. Let L be some fragment of first-order logic and pick some structures A and B so that any sentence of L that is true on A is also true on B. Then the problem with template (A,B) comes from differentiating, for input phi in L, whether A models phi or B fails to model phi, on the promise that one of the two cases holds. If phi is such that A fails to model phi and B models phi, an algorithm can answer anything, it doesn't matter. Actually, the last part of the thesis concerns a slightly more general problem, but let us not worry about this for now.

The thesis is in three distinct parts which appear as three chapters. Each of these chapters corresponds to an article which could be published. Indeed, the first two have been published. Promise problems of the form described have become big business in the mathematical community which studies Constraint Satisfaction Problems (CSPs). Here, the logic L may be taken to be first-order logic restricted to just the existential quantifier and the binary connective conjunction, better known as primitive positive logic. In particular, no negation is allowed. Since the celebrated proofs of the Feder-Vardi conjecture settled the complexity of CSP(A) for all finite A, considerable effort has been put into the corresponding promise problem, which we call the Promise CSP (PCSP). This is a proper generalisation of the CSP, as one may identify CSP(A) with PCSP(A,A). Classifications for PCSP appear to be very tricky: even in the case in which A and B are 2-element structures, there are open cases. This contrasts not just with the CSP, where the 2-element, foundational case was elaborated in the 70s, but also with other generalisations of the CSP. For example, with augmented universal quantification or valued constraints, where the 2-element classification has long been settled. Another strong motivation for studying PCSP comes from approximate graph colouring which may be cast in its decision variant as PCSP(K_i,K_j) in which i \leq j.

Small comments on Introduction.

The writing is generally excellent.

Page 5, line 1. Consider explaining briefly what similar means. This terminology is one of many and not everyone knows it. It is formally defined on page 11, but could have a brief additional explanation here.

Page 5, line 23. Interesting to describe the partial results as "strong". Probably best in case your reviewers are the authors of [BG18].

Page 7, line -8. I don't see any benefit to tt font on Clique. If you use this font here, why don't you use it for other problems like 1-in-3 SAT?

Page 8, line 15. Homomorphism is garbled. (Please run thesis through spell-checker. Perhaps you did but homomorphism isn't in the dictionary.) Line 16. the complexity.

Page 8, Left-hand side restricted PCSP. I suppose the definition here is not fully general. One could have two right-hand structures: D and E, and not just D. Might be worth noting ab initio.

Page 8, Left-hand side restricted PCSP, line 9. the complexity.

Page 8, Thesis outline, line 5. Use `` and '' always (not '' and ''). Later on, you are using the former correctly. This reappears later in the thesis. E.g. on page 9, you use them incorrectly on line 18 (''a lot'') but correctly on line 28 (``the basic tractable cases'').

Chapter 1.

Algorithms for PCSPs tend to come from so-called sandwich structures in the homomorphism order. For finite A and B, the property that all primitive positive sentences being true on A makes them true also on B, coincides exactly with there being a homomorphism from A to B. Now imagine there is some C and homomorphisms from A to C and C to B. Then an algorithm for CSP(C)=PCSP(C,C) solves PCSP(A,B). Indeed, it is not currently known that all tractability of PCSPs cannot be explained this way. The most famous example of such a PCSP is the 2-element case in which A consists of the 1-in-3 relation and B consists of the Not-all-Equal relation. Both CSP(A) and CSP(B) are NP-complete, yet PCSP(A,B) is in P. The canonical C is an infinite structure over Z and it is known that no finite structure C, with CSP(C) in P, exists. This is the result of Barto in LICS 19 [Bar19] and the birth of the notion of finitely tractable (when such a finite C does exist, so this case is not finitely tractable). It is exceedingly natural to seek to extend this result to the scope of [BG18]. This chapter doesn't achieve this, but makes considerable headway by doing it for all of their basic relations. As I said before, results in PCSP seem to require significant work, and the work put into this chapter is already considerable. The proof proceeds by contradiction through an analysis of cyclic and doubly cyclic polymorphisms. The local details to each subcase are rather technical and attest to endeavour commensurate with a doctorate.

Small comments on Chapter 1.

Page 9, line -6. Avoid \neq followed by =. This is horrible notationally. Say instead that \neq defined by \{(0,1),(1,0}\}.

Page 10, Theorem 1(b) compared with (b) later on. Why does the second (b) have an additional subcase?

Page 11. Do you want to say explicitly in Contributions that you classify all the basic tractable cases of [BG18] as finitely tractable or not finitely tractable? At the moment it is a bit vague with "some of the cases of Theorem 1". It only becomes explicit in Section 1.6.

Page 12. Make explicit that you are dealing with the decision version of PCSP in Definition 5. Else you are potentially in conflict with the definition in the introduction.

Chapter 2.

The second chapter concerns the promise version of the model-checking problem for some other fragments of first-order logic. The first comes from allowing only the existential quantifier but both conjunction and disjunction, and the second comes from that after adding universal quantification. Both are deprived of equality and this is important for their generality. Indeed, the CSP is introduced without equality, but it doesn't matter as it can be propagated out. I think this should be noted at the start. The motivation for studying the promise versions of the model-checking problems for these logics is excellent. While with the PCSP it is somewhat tricky to obtain rich classifications, these model-checking problems were quite easy to solve in the non-promise setting. Therefore, they may be more approachable in the promise setting. Indeed, this turns out to be the case. For the first logic, the model-checking classification was very easy, and the promise version is fully classified in this chapter. For the second logic, the model-checking classification was relatively easy, and some strong results have been obtained in the promise setting, in this chapter, but a full classification remains elusive.

Some inspiration has been taken from known results from [Bor08] and [MM18] to develop a Galois theory for these logics in the promise setting. The relevant objects are dubbed (surjective) multi-homomorphisms. (As an aside, it is a travesty that multi replaces the hyper of [Bor08] and [MM18]. Pete Jeavons has used the term multimorphism in VCSP and himself lamented the mixture of Latin and Greek.) Theorems 42 and 48 give the necessary Galois connections for the promise version over the two logics. These theorems contain something subtly new over the non-promise versions in [Bor08] and [MM18]. In the case of the first logic, the full classification is a simple step away. However, for the second logic, only partial progress is made. A key reason for this is that, for both logics, hardness is not attempted a priori, but rather from existing (strong) results for PCSP. This is enough for the first logic, but for the second logic it leaves gaps between NP-hard and Pspace-complete and coNP-hard and Pspace-complete. Interestingly, this are not the only lacuna for the second logic. While the A-shops and E-shops of [MM18] generalise to A-smuhoms and E-smuhoms in this chapter, and AE-shops generalise to AE-smuhoms, the lack of composability causes trouble. While the existence of an E-shop and an A-shop for a template in [MM18] guarantees an AE-shop, this is not the case with E-smuhoms, A-smuhoms and AE-smuhoms. Thus, there appears to be a second gap between in NP and in coNP but not known to be L.

It will be no surprise that this chapter is my favourite, as it extends my own work [MM18]. However, a lot of fine thinking has been done in a short space. Some considerable thought has gone into comprehending [Bor08] and [MM18] and implementing it in the promise setting. The gaps in the classification for the second logic, I find especially tantalising. Marcin Kozic has presented a new and simple proof of the main tetrachotomy of [MM18]. Even with his methods, and much perseverance, I couldn't solve the open questions form this chapter. Indeed, the template that reappears just before Section 2.6 has fascinated us all, and even Dmitriy Zhuk and Tamio-Vesa Nakajima? have failed to prove Pspace-completeness.

Small comments on Chapter 2.

Page 38, line 12. These rules of strict containment restrict the generality of the proof! It is claimed eight lines later that they don't significantly decrease it. This is true depending on taste. I can see that they are used later to simplify some proofs. However, perhaps some comment can be made already here as to how to deal with empty and full relations.

Page 39, line 1. Was surjective already defined in this context?

Page 39, bottom. At this point I was wondering if there had been a global assumption of finiteness for structure domain. I suppose it would be best if there were. I didn't immediately find it in the (global) introduction.

Page 40, Yes-instances and No-instances. I'm not sure why tt font returns here.

Page 41, Definition 30. The stipulations that S is contained in S' and T' is contained in T somehow potentially restrict the generality of this definition. Yet, they are important later. I think it might be worth making a comment on them.

Page 41, line -12. \overline{\mathbb{E}} is not defined. I know it's meaning can be inferred.

Page 43, line -12. This problem c-coloring etc. has no special font at all. I think it might be an idea to make uniform the fonts of problems.

Page 44, Section 2.5.1 line 7. These functions are usually called Skolem functions. Maybe note this.

Page 47, proof of Theorem 50. The second item indeed follows by duality but note that the evaluation is here on B not on A. This is why the third item becomes tricky when A \neq B. At the moment the reader might be confused as to this.

Page 48, line -15 (before the "If there is no such element a_3"). The proof of this case seems not to be concluded.

Page 49, middle. "log-space". Sometimes you write L, sometime logarithmic space. Please avoid too many proliferations.

Page 49, line -18. Maybe say that the mapping from A to B here will be surjective.

Chapter 3

The final chapter considers PCSP "from the other side", that is for left-hand (i.e. input, not left-hand in the promise pair) restrictions. I notice that the right-hand side here is a single structure, not a pair (like in a normal promise problem). It makes sense to study the simpler case first, but perhaps some comment could be made about this. The notion of being "from the other side" comes from Grohe's celebrated result [Gro07] and this is the starting point for the original work in this chapter. Grohe proved that, if C is a recursively enumerable class with bounded arity, then if C does not have bounded treewidth in the core, CSP(C,-) is W[1]-hard. On the other hand it was known that, if C does have bounded treewidth in the core, CSP(C,-) is FPT. Thus, Grohe's result gives a dichotomy, assuming FPT \neq W[1]. Indeed, another way of phrasing it, is that (assuming FPT \neq W[1] and C a recursively enumerable class with bounded arity), CSP(C,-) is in P iff C has bounded treewidth in the core.

The generalisation of Grohe's result obtained in this chapter is significant and technical, but perhaps not so beautiful. It certainly doesn't leave a dichotomy. This chapter has the feeling, unlike the previous two, of a work in progress. Nonetheless, with the right elaboration, it should be publishable. It is certainly a valuable contribution to this thesis.

Small comments on Chapter 3.

Page 53, Section 3.1.1, line 4. Here there is a stipulation of finite set A. Though apparently the signature is now allowed to be infinite.

Page 54, Section 3.1.2, line 10. Space after end of the sentence.

Page 54, line -10. Minors also allow vertex deletion.

Page 56, Section 3.1.4, line 2. an additional parameter.

Page 58, middle. The unordered pair is of *distinct* elements. In a sense the unordered already forces this, but add the word distinct nonetheless.

Page 59, line 7. Normally computing a core is "hard". Maybe remind the reader that here it is "easy", in the sense of fpt.

Page 59, footnote. This is not helpful. No one calls the general homomorphism problem a promise problem and this is because the "promise" (that it is in the correct form) is polynomially computable. Promise problems arise when the promise is somehow intractable.

Page 60, line -2. This recursively enumerating Delta is not an "algorithm", unless you believe RE problems can be solved by algorithms. You don't need that C, D is computable (well, they need not be, but even recursively enumerable), just that they exist.

Page 64, line -15. require a "small number" of homomorphisms from connected A to B.

Evaluation of the corpus.

This thesis contains considerable material, of technical depth and strength warranting the award of a doctorate. I recommend only minor revisions. Please feel free to consider the comments above as not more than advisory where necessary.

Barnaby Martin,
Associate Professor,
Department of Computer Science,
Durham University,
Durham,
DH13LE.
United Kingdom.