

Structural Decomposition Methods: How They Matter (Invited Talk Abstract)

Georg Gottlob

Department of Computer Science, University of Oxford, UK

This talk is based on the published paper [1]. We review structural problem decomposition methods, such as tree and path decompositions. It is argued that these notions can be applied in two distinct ways: Either to show that a problem is efficiently solvable when a width parameter is fixed, or to prove that the unrestricted (or some width-parameter free) version of a problem is tractable by using a width-notion as a mathematical tool for directly solving the problem at hand. Examples are given for both cases, and the refined taxonomy below is illustrated (see [1] for a detailed discussion and examples). When we speak about the treewidth of a problem instance, we mean the treewidth of some graph associated with the instance. Obviously, for each concrete problem, one has to indicate what this graph is, and, whenever necessary, how it can be obtained from the instance.

Taxonomy:

1. **Proving a problem tractable for instances of bounded treewidth.** This is probably the main use of treewidth. The idea is that many practically relevant classes of inputs actually have bounded treewidth, and that for such classes, a polynomial algorithm can be designed. We distinguish between general tractability results and fixed-parameter tractability (FPT) results:
 - 1.a *General tractability results.* We are able to show that the problem becomes tractable, but the best known polynomial algorithms are of complexity $\Omega(n^{f(k)})$, where $\lim_{k \rightarrow \infty} f(k) = \infty$. In many such cases it can actually be proven that the problem is *fixed-parameter intractable* with respect to the treewidth parameter k . A typical problem in this category, is the well-known Constraint Satisfaction problem (CSP). This problem becomes tractable for instances of bounded treewidth or hypertree width.
 - 1.b *Fixed-parameter tractability results.* This is the case if there exists a function f such that the problem is solvable in time $f(k) \times n^{O(1)}$ on instances of treewidth $\leq k$. If the problem is actually solvable in time $f(k) \times O(n)$, then we speak about *fixed-parameter linearity*. In particular, fixed-parameter linearity results can be obtained whenever Courcelle's Theorem can be applied. To illustrate this, we discuss a logical approach to the MULTICUT problem [4].
2. **Using treewidth as a tool for proving a problem to be generally tractable (on all instances).** There are a number of cases where treewidth is used as a tool in general tractability proofs. Again, we may distinguish between two sub-cases:

- 2.a** *Proving that a problem is tractable both for small and large treewidth.* For some problems having some associated implicit or explicit parameter c , it can be shown that there exists a function f such that the problem is tractable both for instances having treewidth $\leq f(c)$ and also for instances having treewidth $> f(c)$. The tractability proof for low treewidth is usually completely different from the tractability proof for high treewidth. As an example for problems in this category, we review the problem of checking whether a loop-free undirected graph has a cycle of length $0 \pmod c$, where c is a fixed constant. This problem was shown to be tractable by Thomassen [6]. This problem actually belongs to a wider class of tractable model-checking problems that can be defined via second-order formulas [3].
- 2.b** *Proving that all yes-instances of a problem have bounded treewidth and that the problem is tractable for this reason.* Here, for some explicit or implicit constant parameter c associated with the problem, one is able to find a function t , such that all yes-instances of the problem must have treewidth $\leq t(c)$. Moreover, one shows that for instances of bounded treewidth the problem is polynomially solvable. Note that this is actually a very special case of Case 2.a. In fact, this means that the high-treewidth case is trivially tractable, because all instances of high treewidth are no-instances. As an example for this category, we will report about results related to the *Partner Unit Problem* (PUP), a problem of industrial relevance [2, 5].

References

1. M. Aschinger, C. Drescher, G. Gottlob, P. Jeavons, and E. Thorstensen. Structural decomposition methods and what they are good for. In T. Schwentick and C. Dürr, editors, *STACS*, volume 9 of *LIPICs*, pages 12–28. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.
2. A. Falkner, A. Haselböck, and G. Schenner. Modeling technical product configuration problems. In *Proceedings of the Workshop on Configuration at ECAI 2010, Lisbon, Portugal*, page 40–45, 2010.
3. G. Gottlob, P. G. Kolaitis, and T. Schwentick. Existential second-order logic over graphs: Charting the tractability frontier. *J. ACM*, 51(2):312–362, 2004.
4. G. Gottlob and S. T. Lee. A logical approach to multicut problems. *Inf. Process. Lett.*, 103(4):136–141, 2007.
5. E. C. Teppan and G. Friedrich. The partner units configuration problem. *CoRR*, abs/1308.6206, 2013.
6. C. Thomassen. On the presence of disjoint subgraphs of a specified type. *Journal of Graph Theory*, 12(1):101–111, 1988.