Appendix A Experiments on Automatic Optimization

In Section 5 we discussed the results of tests over the benchmark suite from the 6*th* ASP Competition. It is worth noting that many domains have been included in several subsequent editions of the ASP Competition series; over the years, the participant teams have iteratively fine-tuned the encodings with the aim of maximizing performance of competing ASP systems. This led, in the case of 6*th* Competition, to a bunch of programs that are already optimized for ASP computation, thus limiting the room for further improvements.

On the one hand, such considerations strengthen the positive results reported in Section 5; on the other hand, one might wonder about what is the impact of the SMARTDECOMPOSITION algorithm when dealing with less refined encodings. We find such inquiry of interest, as it should be in the spirit of the declarative nature of ASP to allow developers to concentrate on knowledge representation rather than on low-level performance issues, that also lead, in some cases, to the production of encodings rather involved and less "human readable". This is why, in addition to the experiments reported in Section 5, we tested our approach also on benchmarks coming from older editions of the ASP Competition series. In particular, we considered the 4*th* Competition, as it is the farthest in time in which encodings comply to the ASP-Core-2 input language standard.

In this set of experiments we measured grounding times of the same three versions of *I-DLV* already taken into account in Section 5, within the same experimental environment (hardware, software, memory and time limits). Table A 1 shows the results; problem names reported in italic denotes domains which are in common between the 4*th* and the 6*th* Competition, while those marked with a '$*$' symbol feature more optimized encodings in the 6*th*. As expected, in this scenario the positive impact of SMARTDECOMPOSITION is even more evident: *I-DLV*$^{\mathrm{SD}}$ grounds a larger number of instances in a significant smaller average time. Intuitively, the less an encoding is fine-tuned, the highest are likely to be the benefits stemming from a careful automatic rewriting of input rules.

Furthermore, for all problems in common between the two ASP competitions herein considered, we tested the systems over the programs obtained by coupling the encodings featured by the 4*th* with the instances featured by the 6*th*, and vice-versa. This should provide us with further information about a the impact of both "manual" optimizations and the ones coming from our automatic method. Intuitively, an encoding may be optimized in different ways, and not necessarily by means of a syntactic modification of rules; for instance, one can push additional information about the domain at hand into the encoding, possibly with constraints, in order to reduce the search space. The results are reported in Table A 2. It is clear that, as one can expect, the best combination is given by *I-DLV*$^{\mathrm{SD}}$ fed with optimized encoding coming from 6*th* Competition. However, some interesting considerations can be made: indeed, in several cases (for instance, *Permutation Pattern Marching*, both over instances from the 4*th*, and, even more, over instances from the 6*th*, that appear to be harder), the smart decomposition guarantees similar or better performance improvements in grounding times that are obtained by the manual tuning.

Appendix B Additional Experiments

We report next the results of an additional experimental evaluation over a further set of benchmark. We take into account problem domains that have been already used for assessing performance of ASP systems in other works; in particular, we considered the domains *Cutedge*, *Graph*

Table A 1: 4*th* Competition – Grounding Benchmarks: number of grounded instances and average running times (in seconds). US indicates that corresponding configurations do not support the adopted syntax.

Problem	**#inst.**	***I*-DLV**		***lpopt* | *I*-DLV**		***I*-DLV^{SD}**	
		#grounded	**time**	**#grounded**	**time**	**#grounded**	**time**
Abstract Dialectical Frameworks	30	30	0.13	30	0.13	30	0.13
Bottle Filling Problem	30	30	4.12	30	6.86	30	4.39
Chemical Classification	30	30	87.81	30	403.38	30	88.22
Complex Optimization *	29	29	36.28	29	38.39	29	36.07
Connected Still Life *	10	10	0.12	10	0.13	10	0.15
Crossing Minimization *	30	30	0.10	30	0.10	30	0.10
Graceful Graphs	30	30	0.37	30	0.39	30	0.37
Graph Colouring *	30	30	0.10	30	0.10	30	0.10
Hanoi Tower	30	30	0.22	30	0.23	30	0.30
Incremental Scheduling *	30	12	297.95	17	229.49	21	221.17
Knight Tour with Holes *	30	20	176.99	20	181.16	20	178.59
Labyrinth	30	30	1.49	30	1.40	30	1.51
Maximal Clique *	30	30	0.34	30	1.11	30	0.34
Minimal Diagnosis *	30	30	2.54	30	2.20	30	2.57
Nomystery *	30	30	34.91	21	100.14	30	35.28
Permut. Pattern Matching *	30	28	57.71	30	3.64	30	62.32
Qualitative Spatial Reasoning *	30	30	2.85	30	2.87	30	2.84
Reachability	30	30	101.93	0	US	30	102.04
Ricochet Robots	30	30	0.27	30	0.31	30	0.31
Sokoban	30	30	2.65	30	2.68	30	2.69
Solitaire	27	27	0.13	27	0.18	27	0.20
Stable Marriage *	30	30	28.35	30	2.65	30	2.46
Strategic Companies	30	30	0.19	0	US	30	0.19
Valves Location	30	30	3.97	30	3.98	30	3.93
Visit-all *	30	30	0.13	30	0.14	30	0.13
Weighted-Sequence Problem *	30	30	2.87	30	9.61	30	2.95
Total Grounded Instances		726/756		664/756		737/756	

5col, *Ground Explosion 2*, *Reach* (Weinzierl 2017) and *TimeTabling* (Perri et al. 2007; Calimeri et al. 2008; Perri et al. 2013).

Table B 1 reports grounding times of the same three versions of *I-DLV* already taken into account in Section 5, within the same experimental environment (hardware, software, memory and time limits). We first note that, regarding *Reach*, even if the problem domain is the same as *Reachability* of Section 5, both encoding and instances are different, as they are taken from (Weinzierl 2017), and do not feature queries. In this case, where decomposition is not applicable because of the rule structure, results show that the use of SMARTDECOMPOSITION, as previously noted, allows us to avoid the overhead due to the invocation of *lpopt*. On the overall, again, one can observe the positive impact of SMARTDECOMPOSITION over grounding times, that allows significant performance improvements in case of *Cutedge* and *TimeTabling*, where *I-DLV*^{SD} times are 97% and 33% lower, respectively, w.r.t. *I-DLV*.

Table A 2: Variation of the encodings – Grounding Benchmarks: number of grounded instances and average running times (in seconds).

Problem	**#inst.**	***I*-DLV**		***I*-DLV^{SD}**			
		#grounded	**time**	**#grounded**	**time**	**#grounded**	**time**	**#grounded**	**time**
4*th* Competition Instances									
		4*th* **Comp. Enc.**		6*th* **Comp. Enc.**		4*th* **Comp. Enc.**		6*th* **Comp. Enc.**	
Incr. Scheduling	30	12	297.95	30	54.65	21	221.17	30	1.93
Maximal Clique	30	30	0.34	30	2.96	30	0.34	30	3.11
Minimal Diagnosis	30	30	2.54	30	1.76	30	2.57	30	1.76
Nomystery	30	30	34.91	30	47.24	30	35.28	30	47.11
Perm. Pattern Match.	30	28	57.71	30	0.27	30	62.32	30	0.27
Stable Marriage	30	30	28.35	30	3.16	30	2.46	30	2.86
6*th* Competition Instances									
		4*th* **Comp. Enc.**		6*th* **Comp. Enc.**		4*th* **Comp. Enc.**		6*th* **Comp. Enc.**	
Incr. Scheduling	20	11	336.77	20	16.07	19	211.61	20	16.21
Maximal Clique	20	20	6.63	20	4.93	20	6.58	20	4.96
Minimal Diagnosis	20	20	4.12	20	5.09	20	4.14	20	4.22
Nomystery	20	20	55.11	20	3.45	20	43.74	20	3.63
Perm. Pattern Match.	20	16	168.93	20	130.47	20	150.99	20	4.21
Stable Marriage	20	0	TO	20	118.55	20	172.68	20	119.53

Table B 1: Additional Grounding Benchmarks: number of grounded instances and average running times (in seconds).

Problem	**#inst.**	***I*-DLV**		***lpopt* | *I*-DLV**		***I*-DLV^{SD}**	
		#grounded	**time**	**#grounded**	**time**	**#grounded**	**time**
Cutedge	130	130	34.38	130	1.64	130	1.08
Graph 5col	180	180	0.12	180	0.14	180	0.12
Ground Explosion 2	17	7	73.33	7	73.02	7	72.87
Reach	50	50	0.29	50	0.76	50	0.30
TimeTabling	27	27	85.57	27	63.67	27	57.67
Total Solved Instances		**367/377**		**367/377**		**367/377**	