# Body-Decoupled Grounding via Solving: A Novel Approach on the ASP Bottleneck*

Viktor Besin[1], Markus Hecher[1], Kaan Unalan[1], and Stefan Woltran[1]

TU Wien, Austria <lastname>@dbai.tuwien.ac.at

## 1 Body-Decoupled Grounding

Motivated by the ASP *grounding bottleneck* [3, 6], the problem of traditional grounding systems resulting in exponentially large programs when instantiating non-ground rules (even for programs with bounded predicate arities), we briefly tease the concept of *body-decoupled* grounding. The idea of this approach is to reduce the grounding size by decoupling dependencies between different predicates of rule bodies. The potential of this is motivated by the following example.

*Example 1.* Assume the following non-ground program $\Pi$ that decides in (1) for each edge ($e$) of a given graph, whether to pick it ($p$) or not ($\bar{p}$). Then, in (2) it is ensured that the choice of edges does not form triangles.

$$p(A,B) \vee \bar{p}(A,B) \leftarrow e(A,B) \tag{1}$$

$$\leftarrow p(X,Y), p(Y,Z), p(X,Z), X \neq Y, Y \neq Z, X \neq Z. \tag{2}$$

The typical grounding effort of (2) is in $\mathcal{O}(|\mathrm{dom}(\Pi)|^3)$. Our approach grounds body predicates of (2) individually, yielding groundings that are linear in the size of the ground atoms. Here, it corresponds to $\mathcal{O}(|\mathrm{dom}(\Pi)|^2)$ due to arity 2.

Based on earlier complexity results for ground and non-ground logic programs [2, 5, 4], we introduce a reduction-based translation from non-ground, tight (and normal) programs to ground, disjunctive programs, an alternative grounding procedure. Our encodings translate a non-ground rule by (i) guessing whether the head atom is part of the answer set, (ii) ensuring satisfiabilty of the rule and (iii) preventing unfoundedness of the guessed head atom. Since every step of the procedure instantiates at most one body predicate at a time, we intuitively deploy body-decoupling, which keeps the grounding size polynomial when assuming bounded predicate arity. Notably, our theoretical results imply that body-decoupled grounding blends-in well with existing approaches, enabling us to interleave different grounding approaches.

## 2 Experimental Results

We implemented a software tool, called `newground`, realizing body-decoupled grounding via search as described above. The system `newground` is written in

* This is an extended abstract of a paper [1] that appeared at IJCAI'22.

Python3 and uses, among others, the API of `clingo` 5.5 and its ability to efficiently parse logic programs via syntax trees. In our implementation, we opted for partial reducability, allowing users to select program parts that shall be reduced and those being (traditionally) grounded, thereby internally relying on `gringo`.

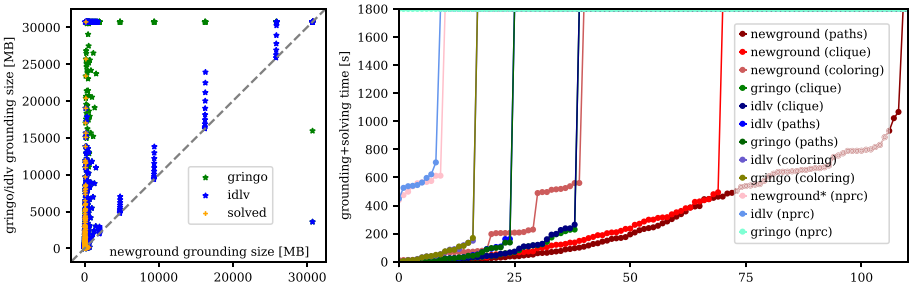


**Fig. 1.** (Left): Scatter plot of grounding size over Scenarios S1–S4 of `newground` (x-axis) compared to both `gringo` (blue) and `idlv` (green) on the y-axis. Those instances that could be solved are highlighted in orange. (Right): Corresponding cactus plot of overall (grounding *and solving*) time over Scenarios S1-S4.

In order to evaluate `newground`, we design a series of benchmarks. Clearly, we cannot beat highly optimized grounders in all imaginable scenarios. Instead, we discuss potential use cases, where body-decoupled grounding is preferrable, since this approach can be incorporated into every grounder. We consider these (directed) graph *scenarios*: (S1) 3-coloring, (S2) reachable paths, (S3) cliques, (S4) non-partition-removal colorings [8] and (S5) stable marriages (ASP comp. 2014), and compared to `gringo` and `idlv` measuring grounding size and (grounding/solving) time.

In terms of grounding time, our experiments show that `newground` in fact outperforms in four of five scenarios. Further, as shown in Figure 1 (left), `newground` also massively reduces the grounding size (almost all dots above diagonal), while keeping instances solvable where `gringo` and `idlv` output groundings beyond 30GB. For the overall (solving) performance we refer to Figure 1 (right). While `newground` performs best, we still see a clear difference between solving and grounding performance (cf. Appendix B), which reveals that only a small amount of those grounded instances can then actually be solved by `clingo` within the remaining time. Detailed plots for each scenario are also in the appendix.

## 3 Conclusion

This work introduces a grounding-approach based on a reduction suggesting the body-decoupling of grounding-intense ASP rules. The reduction translates tight (normal) non-ground rules into disjunctive ground rules, thereby being exponential only in the maximum predicate arity. While our evaluation shows that body-decoupled grounding applied on crucial (tight) program parts reduces grounding size compared to state-of-the-art exact grounders, we are currently working on evaluating and tuning of an implementation for normal programs [7].

## References


1. Besin, V., Hecher, M., Woltran, S.: Body-Decoupled Grounding via Solving: A Novel Approach on the ASP Bottleneck. In: IJCAI'22. pp. 2546–2552. ijcai.org (2022), https://doi.org/10.24963/ijcai.2022/353
2. Bidoít, N., Froidevaux, C.: Negation by default and unstratifiable logic programs. Theoretical Computer Science **78**(1), 85–112 (1991). https://doi.org/10.1016/0304-3975(51)90004-7
3. Cuteri, B., Dodaro, C., Ricca, F., Schüller, P.: Overcoming the grounding bottleneck due to constraints in ASP solving: Constraints become propagators. In: IJCAI. pp. 1688–1694. ijcai.org (2020)
4. Eiter, T., Faber, W., Fink, M., Woltran, S.: Complexity results for answer set programming with bounded predicate arities and implications. Annals of Mathematics and Artificial Intelligence **51**(2-4), 123–165 (2007)
5. Marek, W., Truszczyński, M.: Autoepistemic logic. Journal of the ACM **38**(3), 588–619 (1991). https://doi.org/10.1145/116825.116836
6. Tsamoura, E., Gutiérrez-Basulto, V., Kimmig, A.: Beyond the grounding bottleneck: Datalog techniques for inference in probabilistic logic programs. In: AAAI'20. pp. 10284–10291. AAAI Press (2020)
7. Unalan, K.: Body-Decoupled Grounding in Normal Answer Set Programs. Bachelor's Thesis, Faculty of Informatics, TU Wien, Austria (2022)
8. Weinzierl, A., Taupe, R., Friedrich, G.: Advancing lazy-grounding ASP solving techniques - restarts, phase saving, heuristics, and more. Theory Pract. Log. Program. **20**(5), 609–624 (2020)

## A Additional Examples

*Example 2.* Consider the non-ground program $\Pi$ from Example 1 and facts $\mathcal{F} = \{e(1,2).\ e(2,3).\ e(1,3).\}$. Grounding rule (2) of the program with our reduction $\mathcal{R}(\Pi)$ as described above, results in the ground program $\mathcal{P}$ of Figure 2. Notice that a constraint rule does not need steps (i) and (iii) of the reduction as there are no head atoms being derived. The answer sets of $\mathcal{P}$ restricted to symbol $p$ of $\Pi$ yield $\{\{\}, \{p(1,2)\}, \{p(1,3)\}, \{p(2,3)\}, \{p(1,2), p(2,3)\}, \{p(1,2), p(1,3)\}, \{p(1,3), p(2,3)\}\}$, as expected.

$\mathrm{sat}_X(1) \vee \mathrm{sat}_X(2) \vee \mathrm{sat}_X(3).$ (3)
$\mathrm{sat}_Y(1) \vee \mathrm{sat}_Y(2) \vee \mathrm{sat}_Y(3).$
$\mathrm{sat}_Z(1) \vee \mathrm{sat}_Z(2) \vee \mathrm{sat}_Z(3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \neg b(1).$ (4)
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Y(1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Y(2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Y(3).$ (5)
$\mathrm{sat}_r \leftarrow \mathrm{sat}_Y(1), \mathrm{sat}_Z(1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_Y(2), \mathrm{sat}_Z(2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_Y(3), \mathrm{sat}_Z(3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Z(1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Z(2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Z(3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Y(2), \neg p(1,2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Y(3), \neg p(1,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Y(1), \neg p(2,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Y(3), \neg p(2,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Y(1), \neg p(3,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Y(2), \neg p(3,2).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_Y(1), \mathrm{sat}_Z(2), \neg p(1,2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_Y(1), \mathrm{sat}_Z(3), \neg p(1,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_Y(2), \mathrm{sat}_Z(1), \neg p(2,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_Y(2), \mathrm{sat}_Z(3), \neg p(2,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_Y(3), \mathrm{sat}_Z(1), \neg p(3,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_Y(3), \mathrm{sat}_Z(2), \neg p(3,2).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Z(2), \neg p(1,2).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(1), \mathrm{sat}_Z(3), \neg p(1,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Z(1), \neg p(2,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(2), \mathrm{sat}_Z(3), \neg p(2,3).$
$\mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Z(1), \neg p(3,1).\ \mathrm{sat}_r \leftarrow \mathrm{sat}_X(3), \mathrm{sat}_Z(2), \neg p(3,2).$
$\mathrm{sat} \leftarrow \mathrm{sat}_r.$ (6)
$\mathrm{sat}_X(1) \leftarrow \mathrm{sat}.\ \mathrm{sat}_X(2) \leftarrow \mathrm{sat}.\ \mathrm{sat}_X(3) \leftarrow \mathrm{sat}.$ (7)
$\mathrm{sat}_Y(1) \leftarrow \mathrm{sat}.\ \mathrm{sat}_Y(2) \leftarrow \mathrm{sat}.\ \mathrm{sat}_Y(3) \leftarrow \mathrm{sat}.$
$\mathrm{sat}_Z(1) \leftarrow \mathrm{sat}.\ \mathrm{sat}_Z(2) \leftarrow \mathrm{sat}.\ \mathrm{sat}_Z(3) \leftarrow \mathrm{sat}.$
$\leftarrow \neg\mathrm{sat}.$ (8)

**Fig. 2.** $\mathcal{R}(2)$ with (2) of $\Pi$ from Example 1, guided by our reduction $\mathcal{R}$.

## B Additional Experimental Data and Plots

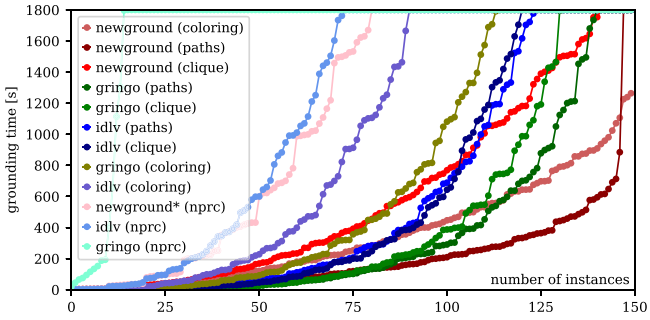


**Fig. 3.** Cactus plot of grounding time over Scenarios S1–S4 for `newground` (red color tones), `gringo` (green tones), and `idlv` (blue tones). The x-axis shows the number of instances; the y-axis is the runtime in seconds, sorted in ascending order for each solver individually. The legend is sorted from best to worst.

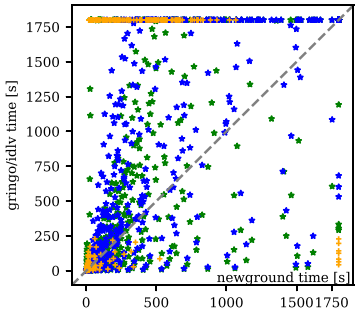


**Fig. 4.** Scatter plot of grounding time over Scenarios S1–S4 of `newground` (x-axis) compared to `gringo` and `idlv` (y-axis); grounding *and solving time* in orange.

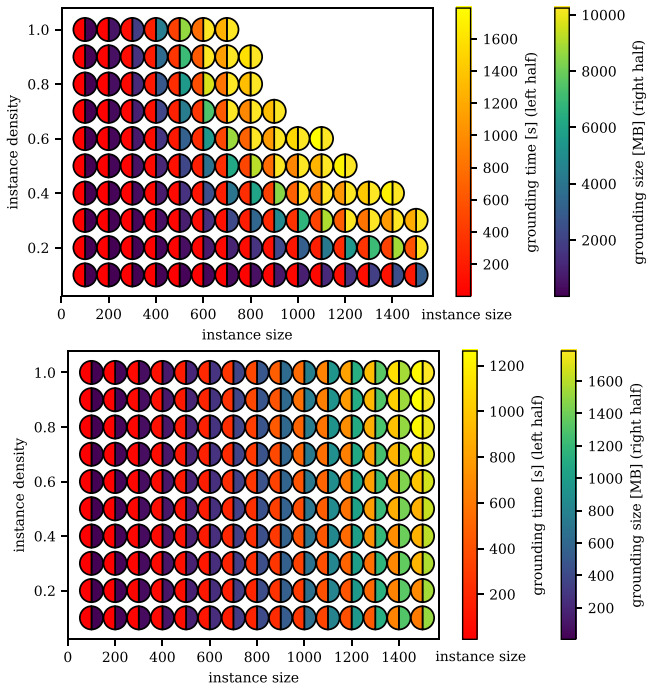


**Fig. 5.** Grounding profile of S1 for `gringo` (top) and `newground` (bottom). The x-axis refers to the instance size; the y-axis indicates density. Circles mark instances grounded $< 1800s$; the left (right) half depicts grounding time (size), respectively. Mind the scales (10000 vs. 1600).

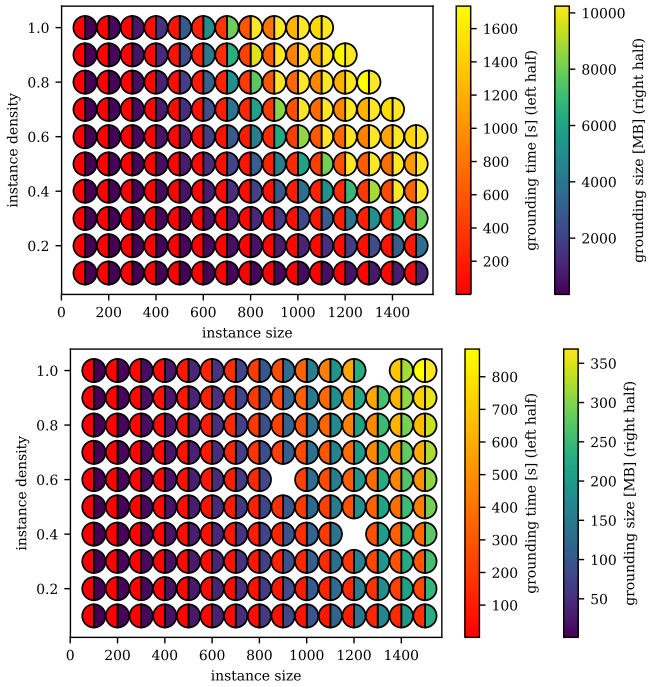


**Fig. 6.** (Top): Grounding profile for `gringo` of S2, similar to Figure 5. (Bottom): Grounding profile for `newground` of S2. Compared to `gringo` (and `idlv`), `newground` grounds larger and denser instances faster, with a grounding size reduction of up to $\frac{1}{50}$.

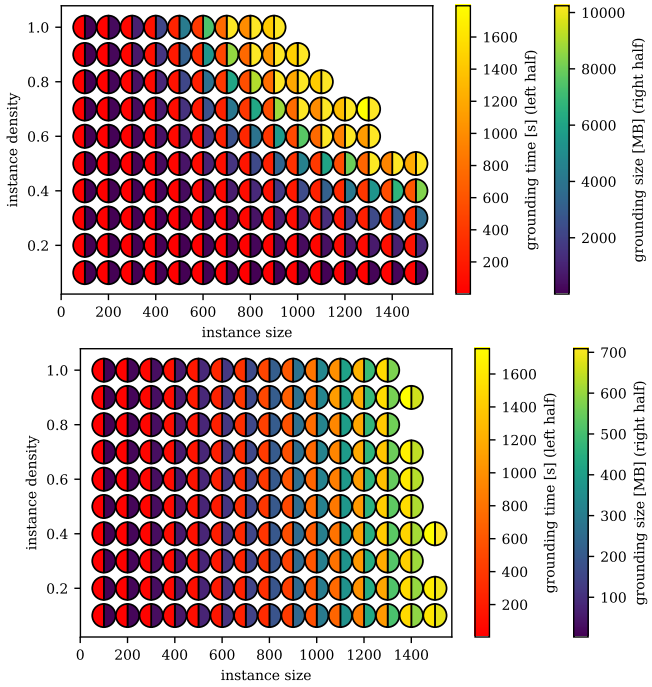


**Fig. 7.** (Top): Grounding profile for `gringo` of S3. (Bottom): Grounding profile for `newground` of S3.

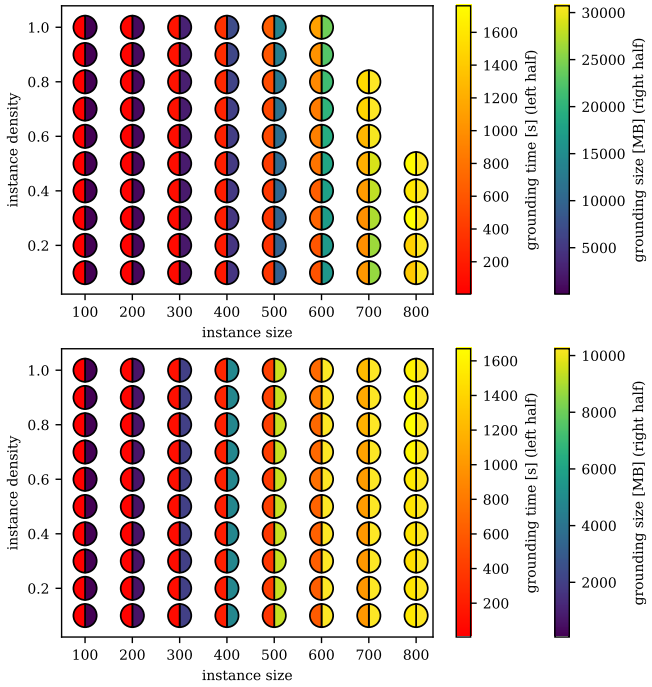


**Fig. 8.** (Top): Grounding profile for `idlv` of S4, which performed better than `gringo`. (Bottom): Grounding profile for `newground` of S4. Both plots depict the same grounding cut-off size of 30GB, but the solvers scale differently.

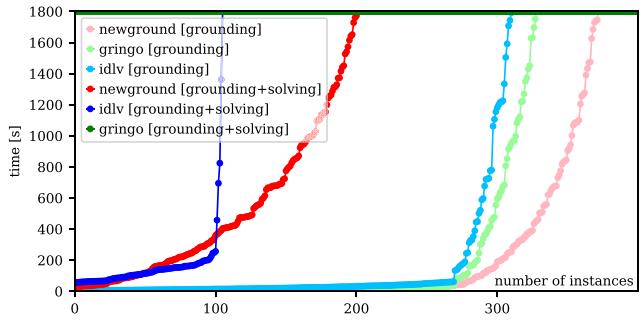


**Fig. 9.** Cactus plot of grounding time as well as grounding and solving performance for Scenario S5 (stable marriage).

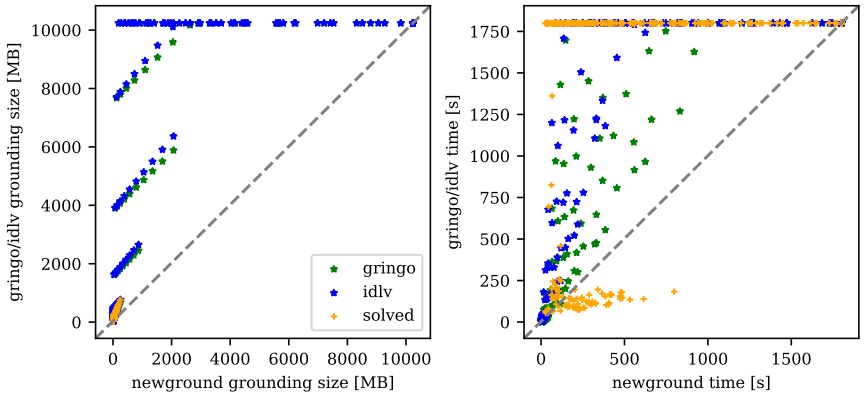


**Fig. 10.** (Left): Scatter plot of grounding size over Scenario S5 (stable marriage) of `newground` (x-axis) compared to both `gringo` (blue) and `idlv` (green) on the y-axis. Those instances that could be solved are highlighted in orange. (Right): Scatter plot of grounding time over Scenario S5 of `newground` (x-axis) compared to `gringo` and `idlv` (y-axis); grounding *and solving time* in orange.