Experiments with Succinct Solvers*

Mikael Buchholtz, Hanne Riis Nielson, and Flemming Nielson

Informatics and Mathematical Modelling, Technical University of Denmark

Abstract. The Succinct Solver of Nielson and Seidl is based on the Alternation-free Least Fixed Point Logic and it is implemented in SML using a combination of recursion, continuations, prefix trees and memoisation. It is known that the actual formulation of the analysis has a great impact on the execution time of the solver and the aim of this note is to provide some insight into which formulations are better than others. The experiments addresses three general issues: (*i*) the order of the parameters of relations, (*ii*) the order of conjuncts in preconditions and (*iii*) the use of memoisation. The experiments are performed for Control Flow Analyses for Discretionary Ambients.

1 Introduction

Static analyses of programs are often constructed as a two-phase process. The first phase extracts sets of constraints from programs and the second phase solves sets of constraints. The benefit of this approach is that the insights and efforts in solver technology may be shared among applications in a variety of programming languages and that it opens up for the use of state-of-the-art tools constructed by experts in their field. The potential disadvantage is that it may be hard to find constraint formats that are sufficiently flexible to be of widespread interest.

As an example, the PAG [5] system (used in the EU-project Daedalus) is targeted toward applications with a fairly static control structure and is not so easy to apply to languages or calculi that have a very dynamic control or mobility structure; however, it does provide means for influencing the performance in the solver in allowing the user to choose between various iteration orders for the iteration-based work-list algorithm around which the system is built and between various data structures for representing abstract domains. A somewhat more flexible tool is the BANE [1] system, originally developed around set constraints but now extended with additional components, including one for type based analyses; however, our experience with the set constraint part shows that there are syntactic limitations to the constraints that can be expressed, and since the actual system has developed beyond the original research papers, it is hard for new users to determine whether these limitations are due to peculiarities of

* This work is partially funded by the SecSafe project (EU IST-1999-29075) and is also published as the SecSafe report SECSAFE-IMM-002-1.0.

the system (and how to overcome them) or whether they are enforced by more fundamental limitations preventing the formulation of analyses that cannot be solved in polynomial time.

Based on recent insights by McAllester [6], and later pursued in collaboration with Ganzinger, we have decided to concentrate on extended formats for Horn clauses. On the one hand this seems to offer adequate flexibility for formulating a number of interesting control flow analyses with applications to security. On the other hand it removes the burden of dealing with a variety of abstract domains — in essence all our abstract domains will be powersets. The actual solver used, the Succinct Solver [8] of Nielson and Seidl, actually implements a rather rich fragment of first-order predicate calculus known as Alternation-free Least Fixed Point Logic; here universal and existential quantifiers as well as disjunctions and stratified negation may be used in preconditions and conjuncts may be used in conclusions.

The Succinct Solver [8] is written in Standard ML and has been coded using a combination of recursion, continuations, prefix trees and memoisation. It incorporates most of the technology of differential worklist solvers and semi-naive iteration developed by Fecht and Seidl and others [3, 4]. However, rather than using an explicit worklist it is based on the top-down solver of Le Charlier and van Hentenryck [2]. This gives a rather robust solver with good time and space performance and whose overall manner of operation has withstood the need for change. Some effort has been spent in "low level" improvements of the code aiming at optimising tail-recursive call etc.

The experience with the Succinct Solver so far suggests that relatively minor changes to the input clauses may affect constant factors rather dramatically (two orders of magnitude) and even the exponent of the complexity polynomial. Hence, the main mode of operation with respect to "optimising" the efficiency of clauses to be solved is to rearrange the clauses before they are submitted to the solver. Indeed, this holds not only for the Succinct Solver but for any off-the-shelf state-of-the-art solver developed using expert insights. (It is thus orthogonal to the approach of the INRIA partner of SecSafe.)

This document gives the preliminary results from our efforts to build up local expertise in operating the Succinct Solver and to perform such minor modifications as are needed to obtain more informative timing measurements[1] and to control the application of the reordering transformation now built into the solver. The main focus of these studies have been on three classes of transformations to the clauses:

[1] As an example, our fine tuning of timing measurements may result in measurements like those of Figure 1 (shown using a linear scale rather than a logarithmic scale as will be the case later); one may observe that the measurements do not fully follow the dotted and predicted curve but seems to lie on line segments that start above the curve and end below the curve; this phenomenon can be explained as duly reflecting the dynamic reorganisation of data structures that takes place inside the Succinct Solver at various points during the solution process.

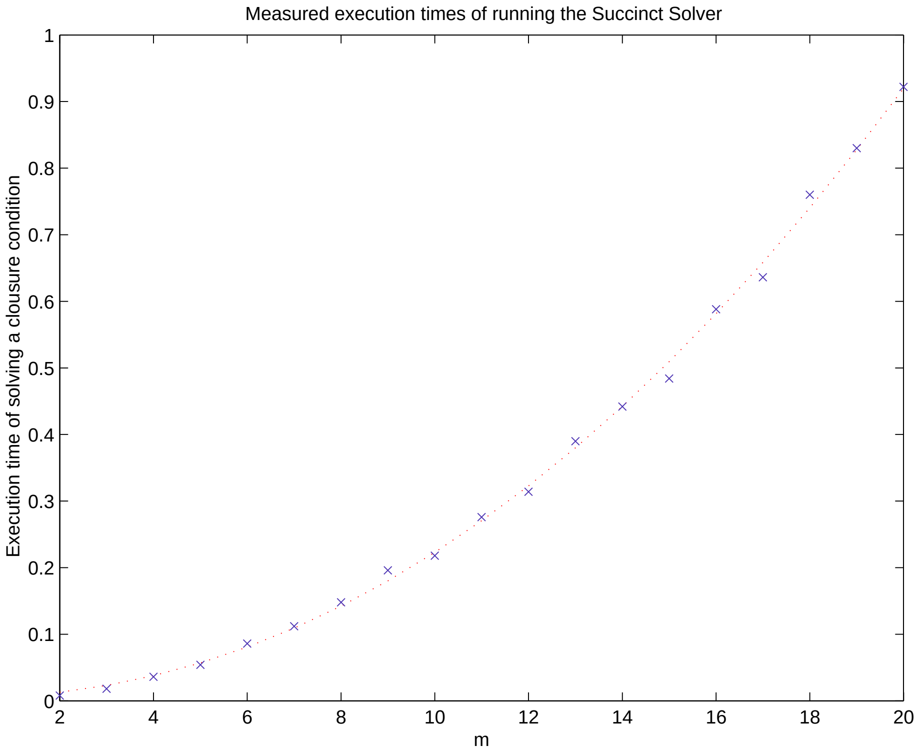


Fig. 1.

1. the order of the parameters of relations,
2. the order of conjuncts in preconditions and
3. the use of memoisation.

Ideally, the experiments should have been performed for the static analysis developed for Carmel but neither constraint generation nor example programs are ready yet. We have therefore focused on a number of Control Flow Analyses for Discretionary Ambients — in keeping with the analyses of Mobile Ambients and the spi-calculus used by Nielson and Seidl when evaluating the performance of the solver.

2 The Test Suite

Discretionary Ambients are presented in [9]. The syntax of processes P and capabilities M are given by the following abstract syntax; here we use n to range over names and μ to range over group names.

$$P ::= (\nu\mu)P \;\mid\; (\nu n : \mu)P \;\mid\; \mathbf{0} \;\mid\; P_1 \mid P_2 \;\mid\; !P \;\mid\; n[P] \;\mid\; M.P$$
$$M ::= \texttt{in}\ n \;\mid\; \texttt{out}\ n \;\mid\; \texttt{open}\ n \;\mid\; \overline{\texttt{in}}_\mu\ n \;\mid\; \overline{\texttt{out}}_\mu\ n \;\mid\; \overline{\texttt{open}}_\mu\ n$$

The construct $(\nu\mu)P$ introduces a new group μ and its scope P; the construct $(\nu n : \mu)P$ then introduces a new name n of the already existing group μ and its scope P. The remaining constructs for processes are as for Mobile Ambients. In addition to the well-known capabilities $\texttt{in}\ n$, $\texttt{out}\ n$ and $\texttt{open}\ n$ we also have the co-capabilities $\overline{\texttt{in}}_\mu\ n$, $\overline{\texttt{out}}_\mu\ n$ and $\overline{\texttt{open}}_\mu\ n$ that in addition to the name n of the object granting the access right also mentions the group μ of the subject that is allowed to perform the operation.

The experiments are carried out on four scalable Discretionary Ambient programs. The first three programs describe a single packet being routed through a network of sites. In the first program $\texttt{s}$-m (for square), the packet is routed through network of $m \times m$ sites named $\texttt{s}_{1,1}, \texttt{s}_{1,2}, \texttt{s}_{2,1}, \ldots, \texttt{s}_{m,m}$ and belonging to different groups $\mathbf{S}_{1,1}, \mathbf{S}_{1,2}, \mathbf{S}_{2,1}, \ldots, \mathbf{S}_{m,m}$. Each site $\texttt{s}_{i,j}$ contains a router table telling where the packet can move next. All router tables have the name $\texttt{r}$ and belong to the same group $\mathbf{R}$. For the site $\texttt{s}_{i,j}$ the router table will instruct the packet to move to either $\texttt{s}_{i+1,j}$ (when $i < m$) or $\texttt{s}_{i,j+1}$ (when $j < m$). Thus, the network topology is as shown in Figure 2 where the arrows indicates the possible movements of the packet.

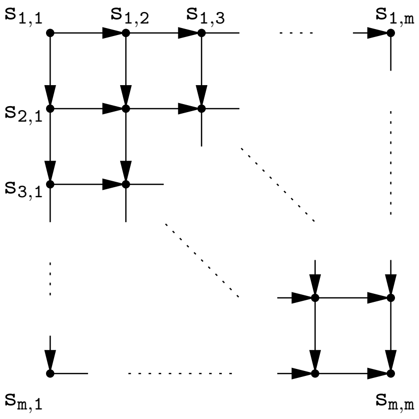


Fig. 2. Network topology of the scalable program $\texttt{s}$-m

In textual form, the program s-2 is shown below

$$
\begin{array}{l}
(\nu \mathtt{p} : \mathbf{P})(\nu \mathtt{r} : \mathbf{R})(\nu \mathtt{s}_{1,1} : \mathbf{S_{1,1}})(\nu \mathtt{s}_{1,2} : \mathbf{S_{1,2}})(\nu \mathtt{s}_{2,1} : \mathbf{S_{2,1}})(\nu \mathtt{s}_{2,2} : \mathbf{S_{2,2}})\\
\quad \mathtt{p}[\mathtt{in}\ \mathtt{s}_{1,1} \mid !\ (\overline{\mathtt{in}}_{\mathbf{R}}\ \mathtt{p}.\ \mathtt{open}\ \mathtt{r})]\\
\mid \quad \mathtt{s}_{1,1}[!\ \overline{\mathtt{in}}_{\mathbf{P}}\ \mathtt{s}_{1,1}.\ \overline{\mathtt{out}}_{\mathbf{P}}\ \mathtt{s}_{1,1}\\
\qquad \mid \quad !\ \mathtt{r}[\mathtt{in}\ \mathtt{p}.\ \overline{\mathtt{open}}_{\mathbf{P}}\ \mathtt{r}.\ \mathtt{out}\ \mathtt{s}_{1,1}.\ \mathtt{in}\ \mathtt{s}_{2,1}]\\
\qquad \mid \quad !\ \mathtt{r}[\mathtt{in}\ \mathtt{p}.\ \overline{\mathtt{open}}_{\mathbf{P}}\ \mathtt{r}.\ \mathtt{out}\ \mathtt{s}_{1,1}.\ \mathtt{in}\ \mathtt{s}_{1,2}]]\\
\mid \quad \mathtt{s}_{1,2}[!\ \overline{\mathtt{in}}_{\mathbf{P}}\ \mathtt{s}_{1,2}.\ \overline{\mathtt{out}}_{\mathbf{P}}\ \mathtt{s}_{1,2}\\
\qquad \mid \quad !\ \mathtt{r}[\mathtt{in}\ \mathtt{p}.\ \overline{\mathtt{open}}_{\mathbf{P}}\ \mathtt{r}.\ \mathtt{out}\ \mathtt{s}_{1,2}.\ \mathtt{in}\ \mathtt{s}_{2,2}]]\\
\mid \quad \mathtt{s}_{2,1}[!\ \overline{\mathtt{in}}_{\mathbf{P}}\ \mathtt{s}_{2,1}.\ \overline{\mathtt{out}}_{\mathbf{P}}\ \mathtt{s}_{2,1}\\
\qquad \mid \quad !\ \mathtt{r}[\mathtt{in}\ \mathtt{p}.\ \overline{\mathtt{open}}_{\mathbf{P}}\ \mathtt{r}.\ \mathtt{out}\ \mathtt{s}_{2,1}.\ \mathtt{in}\ \mathtt{s}_{2,2}]]\\
\mid \quad \mathtt{s}_{2,2}[\overline{\mathtt{in}}_{\mathbf{P}}\ \mathtt{s}_{2,2}]
\end{array}
$$

In the second program denoted lvg-m the packet is again routed through a network of size $m \times m$. Now, there are m router tables $\mathtt{r}_j$ belonging to different groups $\mathbf{R}_j$; the sites $\mathtt{s}_{1,j}, \cdots, \mathtt{s}_{m,j}$ all use the router table $\mathtt{r}_j$. The router table $\mathtt{r}_j$ of $\mathtt{s}_{i,j}$ will instruct the packet to move to *any* of the sites in the row below i.e. one of the sites $\mathtt{s}_{i+1,1}, \cdots, \mathtt{s}_{i+1,m}$ (for $i < m$).

In the third program, called 1-m sites $\mathtt{s}_1, \ldots, \mathtt{s}_m$ of groups $\mathbf{S}_1, \ldots \mathbf{S}_m$ are placed on a line. Now, the router tables $\mathtt{r}$, which are all of group $\mathbf{R}$, instruct the packet to move from the site $\mathtt{s}_i$ to the site $\mathtt{s}_{i+1}$ (when $i < m$).

The final program called sph-m is meant to provoke worst-case behaviour from the analyses. It consists of m ambients (or *spheres*) $\mathtt{s}_1, \ldots, \mathtt{s}_m$ composed in parallel. Each sphere can do *everything* and allow *everything*. That is, a sphere $\mathtt{s}_i$ can move in, move out, and open any other sphere $\mathtt{s}_j$ (including itself) and allows any other sphere to enter, leave, or open it.

2.1 The Analyses

We have experimented with two control flow analyses – a 0-CFA and a 1-CFA. The 0-CFA approximates the behaviour of a process by a *single* abstract configuration that describes all the possible derivatives that the process may have. It amounts to systematically performing the following approximations:

- The analysis distinguishes between the various groups of ambients but not between the individual ambients.
- The analysis does not keep track of the exact order of the capabilities inside an ambient nor of their multiplicities.
- The analysis represents the tree structure of the processes by a binary relation $\mathcal{I}$ modelling the father-son relationship.

Formally, we define the binary relation $\mathcal{I}$ as a mapping

$$\mathcal{I} : \mathbf{Group} \to \mathcal{P}(\mathbf{Group} \cup \mathbf{Cap})$$

where **Group** is the set of groups, and **Cap** is the set of group capabilities and group co-capabilities (i.e. built from groups rather than ambient names). The judgements of the analysis have the form

$$\mathcal{I} \models^{\star}_{\Gamma} P$$

and express that $\mathcal{I}$ is a safe approximation of the configurations that P may evolve into when ambient names are mapped to groups as specified by Γ and when $\star$ is the ambient group of the ambient enclosing the process P. For simplicity, we shall assume that all ambient groups are introduced at the top-level of the program. The analysis is specified in Appendix I.

The 1-CFA follows the same scheme but adds a context by additionally recording information of grand fathers. Thus, the analysis represents the tree structure as a *ternary* relation $\mathcal{I}$ expressing a grand father-father-(grand) son relationship. This is expressed in the ternary relation $\mathcal{I}$ defining the mapping

$$\mathcal{I} : \mathbf{Group} \to \mathcal{P}(\mathbf{Group} \to \mathcal{P}(\mathbf{Group} \cup \mathbf{Cap}))$$

and the judgements of the analysis now becomes

$$\mathcal{I} \models^{\star,\top}_{\Gamma} P$$

As before, $\star$ denotes the group of the ambient enclosing P while, additionally, $\top$ denotes the group of the ambient enclosing $\star$. The specification of the 1-CFA is shown in Appendix IX. The 0-CFA and the 1-CFA have a very similar structure so the effect of similar transformations on the two analyses can easily be compared. The complexity of the 1-CFA is expected to be somewhat higher than that of the 0-CFA.

Representation function and closure condition. In order to use the Succinct Solver, an analysis is split into a representation function and a closure condition. The representation function is responsible for transforming the program into a predicate PRG expressing (an abstraction of) the initial structure of the program; the closure condition then expresses how the relation $\mathcal{I}$ can be computed once PRG is known. The splitting of the initial specification into a representation function and a closure condition is performed automatically. For the 0-CFA, PRG is a binary predicate while it is ternary for the 1-CFA. The results of splitting the analyses can be seen in Appendix II and III, and Appendix X and XI for the 0-CFA and the 1-CFA, respectively. The experiments reported in this paper are concerned with different formulations of the closure conditions for the two analyses.

The relation PRG is given to the solver as a conjunction of ground facts contained in PRG. For the 0-CFA a part of this clause for the program s-2 looks as follows:

$$\begin{array}{l} \ldots \\ \wedge\, \mathsf{PRG}(\mathbf{P}, \mathtt{in}\ \mathbf{S_{1,1}}) \wedge \mathsf{PRG}(\mathbf{P}, \overline{\mathtt{in}}_{\mathbf{R}}\ \mathbf{P}) \wedge \mathsf{PRG}(\mathbf{P}, \mathtt{open}\ \mathbf{R}) \\ \wedge\, \mathsf{PRG}(\mathbf{S_{1,1}}, \overline{\mathtt{in}}_{\mathbf{P}}\ \mathbf{S_{1,1}}) \wedge \mathsf{PRG}(\mathbf{S_{1,1}}, \overline{\mathtt{out}}_{\mathbf{P}}\ \mathbf{S_{1,1}}) \\ \wedge\, \mathsf{PRG}(\mathbf{S_{1,1}}, \mathbf{R}) \wedge \ldots \end{array}$$

Names and sizes. The different formulations of the closure conditions of the 0-CFAs are named `da0_`x, where x is a token that uniquely identifies the analysis. Similarly, the 1-CFAs are named `da1_`x.

In the table below the size of the relations PRG and $\mathcal{I}$ are displayed for the 0-CFA and the 1-CFA. Note in particular that the size of the relation PRG i.e. the size of the program, differs in the in the degree of their polynomial dependency on m for the different programs. The solver works on a universe $\mathcal{U}$ of fixed size. This size is also displayed in the table.

	0-CFA			1-CFA		
	Size of $\mathcal{U}$	Size of PRG	Size of $\mathcal{I}$	Size of $\mathcal{U}$	Size of PRG	Size of $\mathcal{I}$
1-m	$6m+7$	$6m+2$	$9m+3$	$6m+8$	$8m-5$	$2m^2+17m-1$
s-m	$6m^2+7$	$6m^2+2$	$9m^2+3$	$6m^2+8$	$9m^2-2m-1$	$2m^4+18m^2-2m$
lvg-m	$6m^2+5m$ $+7$	$2m^3+5m^2$ $+5$	$2m^3+8m^2$ $+2m+6$	$6m^2+5m$ $+8$	$5m^3+3m^2$ $-3m+3$	$2m^4+m^3+12m^2$ $-2m+6$
sph-m	$3m^2+5m$ $+1$	$6m^2+m$	$3m^3+4m^4$ $+m$	$3m^2+5m$ $+2$	$6m^2+1$	$3m^4+7m^3$ $+4m^2+m$

Table 1.

2.2 Timing the Experiments

The time the solver uses to calculate the analysis result is measured as the CPU-time used by the SML interpreter, which runs the solver. The execution time is split into two contributions: the time for the initialisation phase and the time for solving the constraints.

The first contribution includes the time it takes to load analysis clause files, generate the representation relation from the Discretionary Ambient program and initialise all the data structures in the solver. Additionally, in the initialisation time we include the time it takes to "solve" the clause giving by the representation relation PRG. This clause consists only of ground facts so solving it simply means that the content of PRG is inserted into the data structure that the solver uses to represent relations. The second contribution of the measured execution time comes from solving the closure condition. We measure all execution times both with and without the time used for garbage collection. Execution times including time spent on garbage collection show great variation when the same experiment is performed several times. Therefore, we will only comment on execution times where the time spent on garbage collection is not included.

Estimating Parameters in Execution Times. Having measured the execution time of the solver it is of interest to see how it relates to the size of the program. Here, we are interested in the execution time spent on solving the closure condition.

We know that execution time of the solver has a polynomial upper bound, but we are hoping to give a more precise description of the measured times. Knowing the solver algorithm, it should be possible to explain execution times from knowledge of the size of the clause, the size of the universe, and the size of the computed relations. For our examples, the last two sizes can be computed from the size m in example programs (as shown in Table 1). The size of the clause for the closure condition is small compared to this. Therefore, we assume that the measured execution times t may be explained by the model

$$t = c_1 \cdot m^x + c_0$$

for $x \in \mathbb{R}$. In order to estimate c_0, c_1 and x, the measured times are fitted to this model by a least-square fit for different values of x. The best fit is chosen as the one where the 2-norm of the difference between the model and the measured execution times are the smallest, i.e. where the error is smallest in a least-square sense.

As we consider initialisation time separately the constant c_0 is expected to be 0 since, when $m = 0$ no time is used. For all our experiments we have that c_0 is in the order of the accuracy of the measurement, so we will not comment on it further.

This estimation method is somewhat unstable, which for some part may be explained by the fact that the measured times are not entirely explained by the proposed model. For example, there may be considerable contributions which are explained by terms in the complexity polynomial of lower degrees. Therefore, the estimation results should only be seen as a guideline for further analysis of the measured data.

3 The Order of the Parameters of Relations

The Succinct Solver uses prefix trees as an internal representation of relations, which together with an number of optimisations will give a faster algorithm. In this section we will investigate the effect of specifying clauses to utilise these optimisations.

The optimisations using prefix trees may be explained from what happens when a k-ary relation $R(x_1, \ldots, x_k)$ is checked in a precondition. Whenever a precondition is checked some of the variables $x_1, \ldots, x_n$ will a priori be bound to certain values. These variable bindings are recorded in the environment η. The prefix implementation utilise that some maximal prefix $p = x_1, \ldots, x_i$ for $i \leq k$ of the variables may be bound in η. The optimisation happens at the following two places when a relation R is checked in the precondition

1. Unification is made between the environment η and all the elements already in the relation R. However, if we know that a certain prefix p of variables are bound in η then we also know that unification will fail for elements with a prefix that is different from p. Therefore, unification only needs to be done with elements in the relation that has a matching prefix, thus, saving work.
2. Whenever the relation R is checked in a precondition a *consumer* is registered. The consumer serves as a reminder of the check made in the precondition. If a new element is added to R the consumer will be activated so the check can be performed for *this* new element. Unification should only be done for elements with prefixes that match the bound variables in the precondition (see 1.), so consumers are stored with the knowledge of these prefixes and only activated when prefix-matching elements are added.

The prefix optimisation result in higher efficiency the longer the prefix p is. Taking advantage of this optimisation, therefore, amounts to specifying relations in the preconditions in such a way that the prefixes of bound variables become as long as possible.

For example, assume that the relation $R(x_1, x_2)$ appears in a precondition where x_2 is always bound and x_1 is always unbound. Then, the prefix of bound variables has length 0. Alternatively, one could query the *inverse* relation $R'(x_2, x_1)$ where the parameters are swapped. This gives a prefix of length 1 and is, thus, preferable over querying R. Now, suppose that a specific clause only has queries to the relation R so that R' is the best choice[2]. Then R could be substituted with R' everywhere in the clause. Otherwise, it may be better to include both R and R' though this will double the space used to store the relation and add extra work in order to keep the two relations consistent.

We have conducted experiments using a variation of the analysis `da0_2` on the program s-*m*. This analysis is called `da0_6` and contains the two relations PRG and $\mathcal{I}$ [3]. We have made a number of experiments using different combinations of the order of the parameters of these two relations. In order to do that, the automatic reordering of parameters, which is implemented in the solver, has been disabled. First, we have made experiments using only a single copy of each of the relations by manually restating the analysis with parameters of one or more relations swapped:

`da0_6fifi`: $\mathcal{I}$ in the order (father, id). PRG in the order (father, id).
`da0_6fiif`: $\mathcal{I}$ in the order (father, id). PRG in the order (id, father).
`da0_6iffi`: $\mathcal{I}$ in the order (id, father). PRG in the order (father, id).

[2] The relation R' is the optimal choice if it is queried when either x_2, both x_1 and x_2, or none of the variables are bound.

[3] The clause describing the analysis `da0_2` uses terms, which internally in the solver are translated to auxiliary relations before the clause is solved. Thus, the clause contains 7 auxiliary relations when it is solved. In `da0_6` we have formulated the analysis so there is no gain from reordering parameters in these relations. Thus, we disregard effects from reordering parameters in the auxiliary relations.

da0_6ifif: $\mathcal{I}$ in the order (id, father). PRG in the order (id, father).

An example of these closure conditions are shown in Appendix VII. Second, we have conducted an experiment allowing *multiple copies* of the same relation using the *automatic* reordering implemented in the solver.

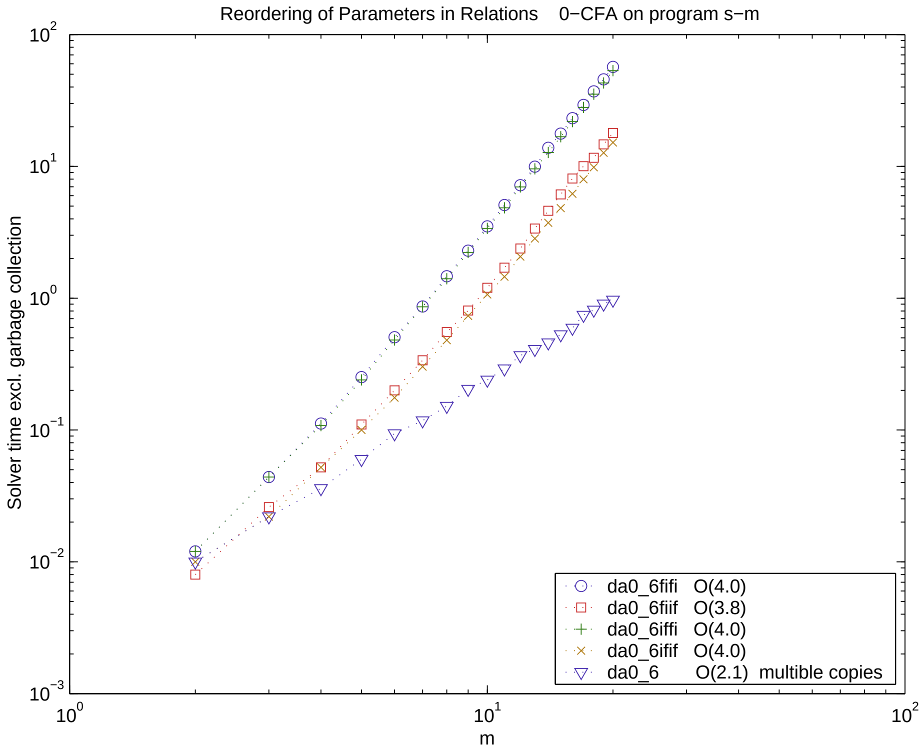


Fig. 3.

In Figure 3, the measured execution times of running the analysis `da0_6` with parameters of the relations in different orders are shown in relation to m; the size of the network. The plot is made using logarithmic scales on both axes. Thus, a polynomial dependency will result in a straight line. Higher degrees of the polynomial will appear as a steeper gradient. Hence, two polynomials with the same degree will appear as parallel lines. The polynomial with the smallest coefficient will appear as the lowest of the two lines.

We see from Figure 3 that reordering parameters in $\mathcal{I}$ or PRG give improvements, which are a constant factor better than the original order `fifi`. Using the order (id, father) of the relation PRG gives significant improvements, while changing the order of the parameters of $\mathcal{I}$ is less significant. However, when we use multiple copies of the relations the degree of the complexity polynomial drops by 2 and, thus, gives a significant improvement.

Along the lines of the experiments with the 0-CFA we have conducted experiments with the 1-CFA. Again, we have defined an analysis `da1_6`, which this time is based on the analysis `da1_2` with an example given in Appendix XIV. Here, `g` in the suffix of an analysis name stands for *grand father*. This time, there are 36 $(3! \cdot 3!)$ permutations of the order of the parameters in the two ternary relations $\mathcal{I}$ and PRG. To reduce the amount of data we have fixed the order of parameter in PRG to (id, grand father, father) and varied the order of parameters in $\mathcal{I}$.

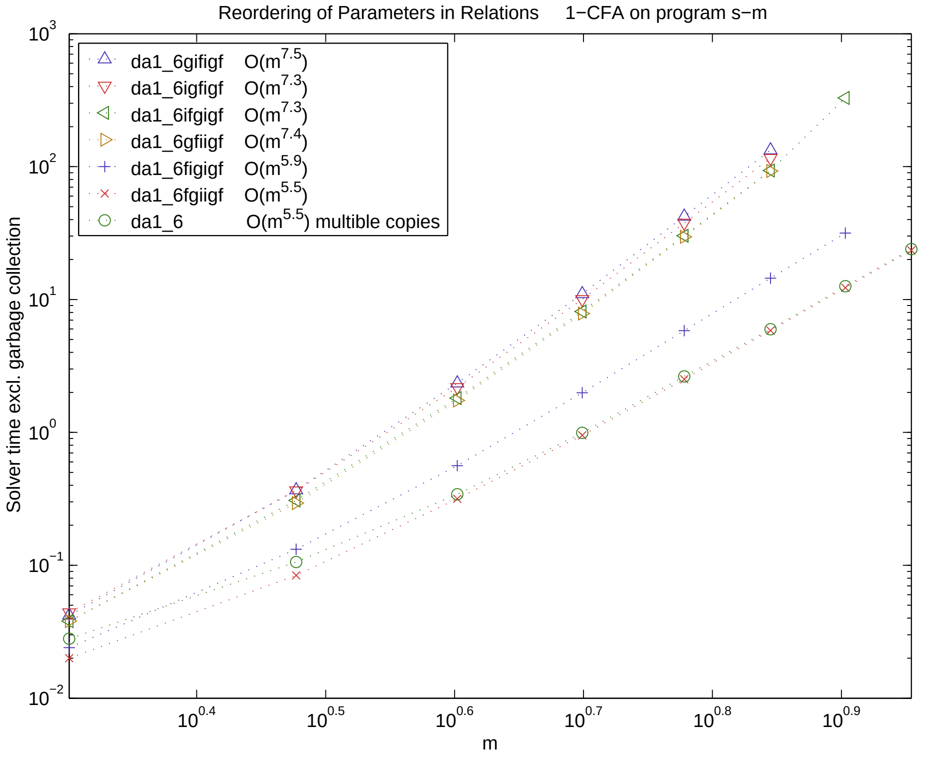


Fig. 4.

The results of these experiments are shown in Figure 4. Again we see that changing the order of parameters may change the complexity – even by a degree in the complexity polynomial. The last analysis, `da1_6`, contains two copies of the relation $\mathcal{I}$ generated by the reordering strategy implemented in the solver. One copy of the relation has the parameters in the order (father, grand father, id) while the other copy has the order (father, grand father, id). However, this analysis is only as good as the analysis that contains only a single copy of $\mathcal{I}$ with parameters in the order (father, grand father, id). Thus, in this case, there is effectively no improvement gained from copying the relation, though it yields longer prefixes of bound variables. Moreover, copying the relation will require additional space to store the relations.

To summarise, we have given evidence that the strategy for reordering parameters, which is implemented in the Succinct Solver will increase efficiency. The increase is often by degrees of the complexity polynomial! However, we have also given an example where the reordering did not increase efficiency even though the solver used two copies of the same relation. We conclude that the reordering strategy implemented in the Succinct Solver, in general, is highly recommendable but there are cases where the same speed can be achieved without copying the relations. With care, this may be used when fine tuning clauses for space consumption. The experiments done in the remainder of this paper have been performed using the reordering strategy implemented in the Succinct Solver.

4 The Order of Conjuncts in Preconditions

Preconditions are evaluated from left to right and in the context of an environment η. When checking a query to a relation R the evaluation of the remainder of the precondition is performed for all the new environments η', which are made by unifying η with an element currently in R. The unification will fail when the binding of the variables in η does not coincide with the element of R. In this case, no further work is done. Thus, we may expect to gain efficiency by making the unification fail as early as possible in the evaluation of a precondition. This is the objective of the experiments made in this section.

Now, the question is how to make unification fail early and consequently propagate as few environments as possible. As an example, consider the clause

$$\forall x\ [R(x, a) \wedge R(b, x) \Rightarrow Q(x)]$$

where a and b are constants. Initially, the query $R(x, a)$ is evaluated and unification is performed with every element in R. Since x is unbound, the unification succeeds for the elements in R, which have a as the second component. For each of these environments $R(b, x)$ is evaluated.

Now, suppose we have a priori knowledge that the relation R will contain few elements with b as the first component but many elements with a as the second component. In this case, swapping the conjuncts in the precondition, i.e. the clause

$$\forall x\ [R(b, x) \wedge R(x, a) \Rightarrow Q(x)]$$

will be more efficient as fewer environments are propagated from the first query to R. This observation leads to the general optimisation strategy that putting queries, which restrict the variable binding most, at the front of preconditions will increase efficiency. Note that this strategy may require a priory knowledge of the content of the relations and this knowledge may not always be available.

For our analyses, we do in fact have some a priori information on how the relations are built. Consider for example the clause for checking capabilities `in` μ in the analysis `da0_1`:

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^p, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right]$$

Here, the inner precondition $(\mathcal{I}\langle \mu^a, t_1 \rangle \wedge \mathcal{I}\langle \mu^p, \mu^a \rangle \wedge \ldots)$ consists of queries to $\mathcal{I}$ and the explicit binding of a term to t_2. The first query, binds the father μ^a of the capability, the second and the third query finds a path to the sibling, while the last two queries find co-capabilities. Each of these queries will give rise to a different number of environments to be propagated as a result of successful unification. For example, we may expect that there are only a few co-capabilities matching the μ in the capability – compared to the number of sibling ambients of a given ambient. Therefore, it will be a good idea to move the query concerning co-capabilities to the front of the precondition.

We have specified three analyses (`da0_1`, `da0_2` and `da0_3`) using different orders of conjuncts in the inner preconditions for all three capabilities in, out, and open. Additionally, we have specified a fourth analysis, where the variables t_1 and t_2 used to match terms are bound *before* the explicit unification of the term such as $\overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2$.

In summary the analyses are:

`da0_1`: first recognise a capability, then the path to the root of the redex and finally the matching co-capability (see Appendix III).

`da0_2`: first recognise a co-capability, then the path to the root of the redex and finally the matching capability (see Appendix IV).

`da0_3`: first recognise a co-capability, then the matching capability and finally the path to the root of the redex (see Appendix V).

`da0_7`: as `da0_2` but explicit unification of terms are done after the binding of the term variable (see Appendix VIII).

Figure 5 shows the result of running the solver using the four analyses. We see that `da0_3` is a constant factor better than `da0_1`. For `da0_2` the polynomial degree of the complexity is changed and, thus, yields a significantly better analysis. `da0_7` is only a constant factor worse than `da0_2`.

We have manually calculated the number of environments which will be generated for the three analyses and confirmed the empirical results. However, these calculations rely heavily on the structure of the program `s`-m. Therefore, the improvements we observe may only apply for precisely these programs and need not be valid in general. Hence, the experiments have been repeated for the pro-

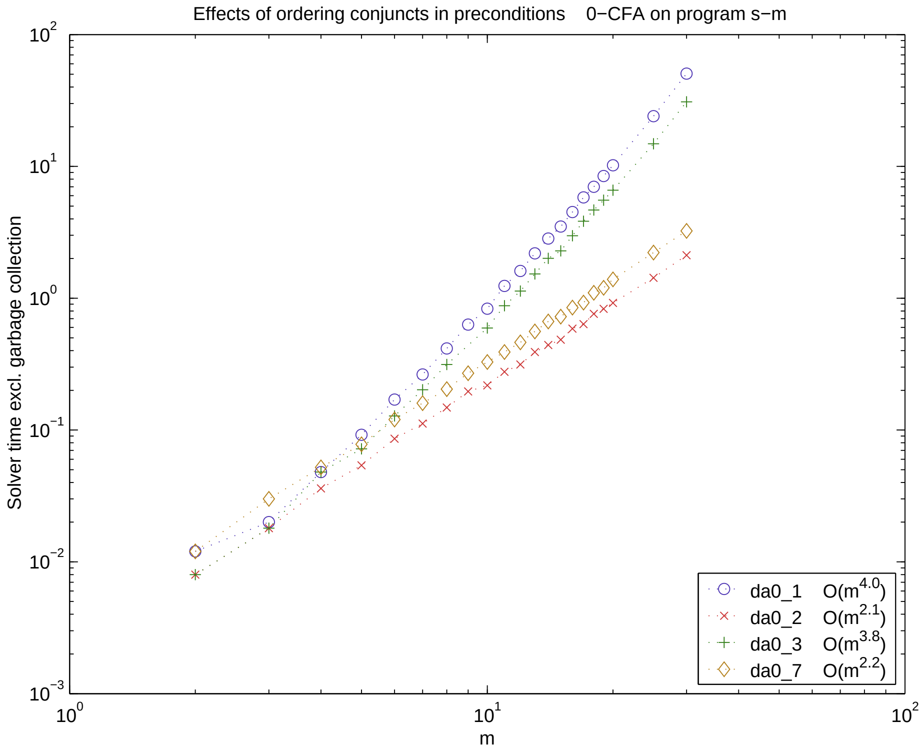


Fig. 5.

grams 1-m, lvg-m and sph-m. The results can be found in Appendix XXIV. For 1-m we see a pattern identical to the one found for s-m. For lvg-m we also observe something similar, though the change of degree is not as significant. Thus, we conclude that the effect of reordering the conjuncts in the preconditions is advantageous regardless of the network topology, which differs in the three programs 1-m, s-m, and lvg-m. However, our argument for putting co-capabilities at the front was that there was only a few of these which would match a given μ. This is *not* true in general. One example where it does not hold is the programs sph-m. Consequently, this reordering of the conjunct should have no effect, which is also confirmed experimentally as shown in Appendix XXIV. We conclude that the programming style in this case influence the effect that our proposed transformation has on efficiency of solving our analyses.

Similar experiments have been conducted for the 1-CFA by testing the analyses da1_1, da1_2, da1_3, and da1_7 which may be found in Appendix XI through XV. The result of the experiments on the program 1-m is shown in Figure 6 while experiments with the programs s-m, lvg-m, and sph-m are shown in Appendix XXV. Figure 6 shows the general pattern of these experiments. The complexity

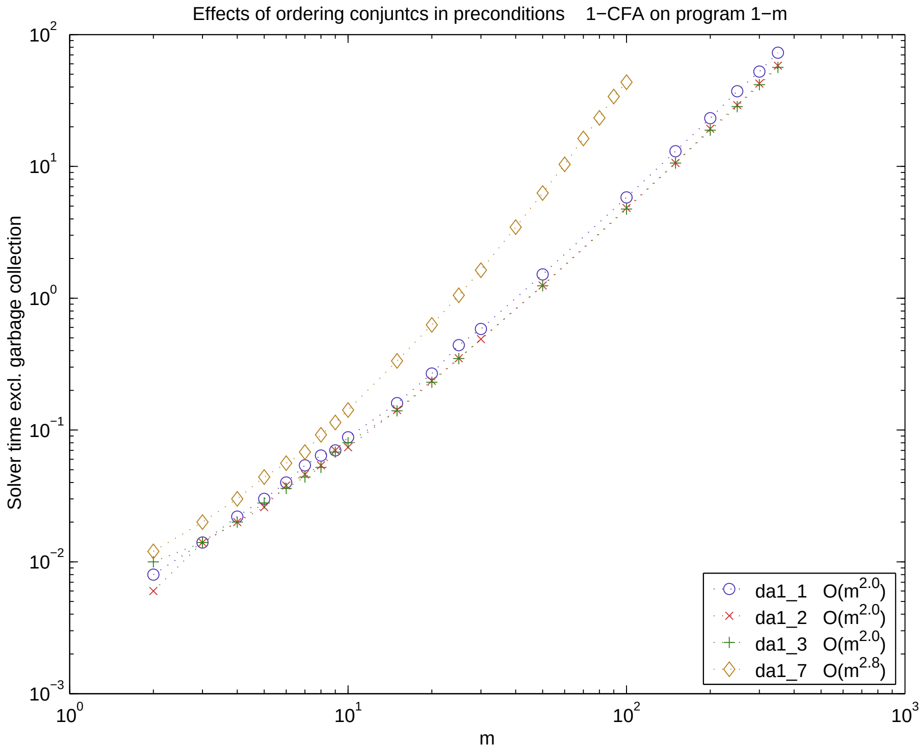


Fig. 6.

of solving the analyses `da1_1`, `da1_2`, `da1_3` differs only by a constant factor. The analysis `da1_7`, on the other hand, is worse by degrees of the complexity polynomial.

In summary, we have shown that the order of conjuncts in preconditions has a significant effect and that the effect may be explained by the number of environments propagated through the preconditions. Almost everywhere we have been able to modify the order of conjuncts so the degree of the complexity polynomial changes. However, we have no clear indication that there is particular type of rewriting, which will *always* change the degree of the solving time complexity. Yet, a particular analysis may be beneficial for programs using a specific programming style.

5 The Use of Memoisation

Extra work arising from needless propagation of environments in the preconditions may also arise because the precondition uses variables not relevant for the conclusion. Suppose we have a clause as

$$\forall x, y_1, y_2, y_3\ [P(x, y_1) \land Q(y_1, y_2, y_3) \Rightarrow R(y_1, y_2, y_3)] \tag{1}$$

All the environments built when checking $P(x, y_1)$ will contain x. These environments are propagated although x is not used in neither $Q(y_1, y_2, y_3)$ nor in $R(y_1, y_2, y_3)$. If there are two of these environments where x is different but y_1 has the same value, the evaluation of $Q(y_1, y_2, y_3)$ and $R(y_1, y_2, y_3)$ will be done twice, although the result will be the same. In this section we investigate and compare two approaches, which target this problem.

5.1 The Virtues of Tiling

Tiling is a systematic transformation on Horn Clauses proposed in [7]. The transformation may improve worst-case complexity of solving Horn Clauses by reducing quantifier nesting depth if there exists queries in a precondition with variables that are not used in the conclusion. Hereby, it also targets the problem described above.

As an example of tiling, we take the clause (1) from above. Applying the tiling transformation of the variable x results in the new clause

$$\begin{array}{l} \forall x, y_1\ [P(x, y_1)] \Rightarrow S(y_1) \land \\ \forall y_1, y_2, y_3\ [S(y_1) \land Q(y_1, y_2, y_3)] \Rightarrow R(y_1, y_2, y_3) \end{array}$$

thus, introducing an auxiliary relation S, which allows the second clause to "ignore" the different values of x. In general, tiling gathers *all* queries involving the candidate variable and builds an auxiliary relation containing the values of the remaining variables, which have a common candidate variable. Furthermore, all the queries involving the candidate variable (above only $P(x, y_1)$) is removed from the original clause and a query to the auxiliary relation is inserted ($S(y_1)$ in the example).

The tiling transformation disposes of the unnecessary propagation of environments, which only differs in the candidate variable. Thus, the benefits of tiling will (at best) be proportional to the number of such environments. However, by applying the tiling transformation there is a risk of building *very large* auxiliary relations! These relations will both take up memory and require time to compute, thus, with the danger of nullifying the gain or even worsening the overall performance.

Horn Clauses and sharing. Tiling is a transformation that works on Horn Clauses, which is a subset of Alternation-free Least Fixed Point Logic; the clause format of the Succinct Solver. However, transforming an Alternation-free Least Fixed Point Logic formula into a Horn Clause may alter the complexity of solving the clause. For our analyses, the only change in complexity is that we lose the positive effects of *sharing* preconditions between multiple conjuncts in the conclusion as illustrated by the transformation of a) into b) below

$$\text{a)}\quad pre \Rightarrow con_1 \wedge con_2 \qquad\qquad \text{b)}\quad \begin{array}{l} pre \Rightarrow con_1 \wedge \\ pre \Rightarrow con_2 \end{array}$$

This transformations will, in theory, only make the clause b) a constant factor worse than the clause a) and the size of the constant will depend on the size of the precondition *pre*. For small preconditions this effect does, however, not show up in empirical data (see e.g. the difference of running `da0_2` and `da0_t1` on Figure 7). For larger preconditions, such as the ones that may be found in the 1-CFA in the inner precondition of "capability clauses", the effects are measurable (compare e.g. `da1_2` and `da1_t1` on the figures in Appendix XXVIII).

Experiments with tiling. We have transformed the 0-CFA analysis `da0_2` into Horn Clause and performed tiling of different candidate variables. The resulting analyses, of which examples are shown in Appendix XVI through Appendix XIX, are summarised below

`da0_t1`: Horn Clause form of the analysis `da0_2`
`da0_t2`: $\star$ tiled from `da0_t1`
`da0_t3`: t_1 tiled from `da0_t2` (i.e. $\star$ then t_1 are tiled – in that order)
`da0_t4`: t_2 tiled from `da0_t3` (i.e. $\star$, t_1 then t_2 are tiled – in that order)
`da0_t5`: t_1 tiled from `da0_t1`
`da0_t6`: t_2 tiled from `da0_t1`
`da0_t7`: μ or μ^p tiled from `da0_t1`

Figure 7 shows the result of the different tiling transformations. Tiling improves the degree of the complexity polynomial for the transformations `da0_t2`, `da0_t3`, `da0_t4` i.e. the transformations where the variable $\star$ is the first one that is tiled.

Tiling of t_2 (`da0_t6`) has no effect. The variable t_2 is used to bind co-capabilities. However, for the programs lvg-m every instance of a co-capability will only appear inside *one* parent ambient. Thus, queries, which are on the form $\mathcal{I}(\mu, t_2)$, will only result in propagation of one environment for each t_2 so the tiling will have no benefits. Additionally, the auxiliary predicate generated by tiling does not cause substantial amounts of additional work.

The two analyses `da0_t5` and `da0_t7` are examples of tiling transformations that increase the work of solving the clause. Here, the auxiliary relations introduced by tiling become so large that the work of calculating these relations overshadows the work of solving the rest of the clause. When `da0_t5` is run on the program s-m, as shown in Appendix XXVI, the analysis is, however, as good as the original analysis. This is because there are no single capability that occurs more than once in the program s-m. Thus, the auxiliary relation (HasIn), which is introduced by tiling, does not explode in size for this program. Consequently, the analysis is no more expensive to solve than the original one.

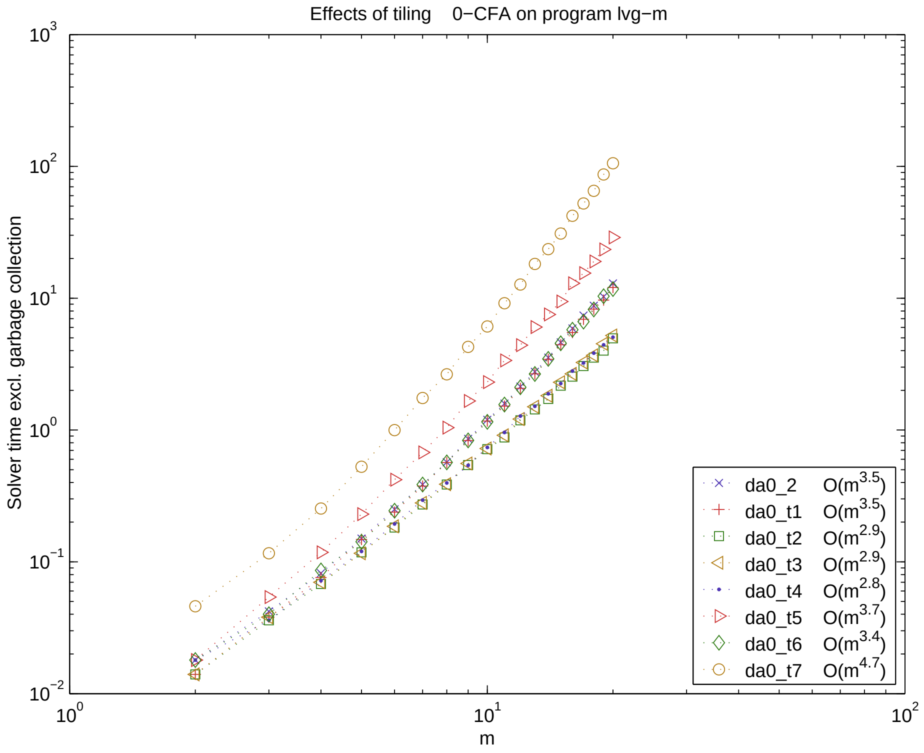


Fig. 7.

Similar arguments can be given when testing the analyses on the remaining test programs as shown in Appendix XXVI. Some of the tilings have, additionally, been applied to the 1-CFA as shown in Appendix XXVIII and this give similar results.

5.2 Memoisation

The Succinct Solver includes a build-in facility that avoids propagation of identical environments by applying memoisation techniques. Propagation of *completely* identical environments can only occur at disjunctions and at existential quantifications in a precondition so memoisation is only applied for these constructs.

However, a similar phenomenon can occur at other places. Consider e.g. the clause (1) on page 16. Here, several environments that differ only in the value of x may be propagated. These are *essentially* identical but since neither existential quantification nor disjunction is used we cannot (directly) benefit from the memoisation techniques implemented in the solver.

Instead of solving the clause (1) we may transform it into the logically equivalent clause

$$\forall y_1, y_2, y_3\ [[\exists x\ [P(x, y_1)] \wedge Q(y_1, y_2, y_3)] \Rightarrow R(y_1, y_2, y_3)] \tag{2}$$

Here, the universal quantification of x has been transformed into an existential quantification in the precondition. The basic solving algorithm of the Succinct Solver will solve the two clauses (1) and (2) in the same way [4]. However, for the clause (2) the memoisation scheme ensures that no identical environments are propagated.

The replacement of universal quantification by existential quantification in preconditions can be applied as a general transformation whenever the quantified variable does not occur in the conclusion. Basically this addresses the same problem as the tiling transformation. Therefore, we may expect a similar behaviour when solving the result of the two transformations.

Experiments with memoisation. Again, we have transformed the analysis into Horn Clauses. From this analysis we have applied the transformations, which inserts existential quantifiers. The analyses using memoisation are tailored to resemble the analyses used to test the tiling transformation. Examples of the analysis are shown in Appendix XX through XXIII. In summary they are

`da0_t1`: Horn Clause form of the analysis `da0_2`
`da0_m2`: $\star$ existentially quantified from `da0_t1`
`da0_m3`: $\star$ and t_1 existentially quantified from `da0_t1`
`da0_m4`: $\star, t_1$, and t_2 existentially quantified from `da0_t1`
`da0_m5`: t_1 existentially quantified from `da0_t1`
`da0_m6`: t_2 existentially quantified from `da0_t1`
`da0_m7`: μ or μ^p existentially quantified from `da0_t1`
`da0_m9`: all possible variables existentially quantified from `da0_t1` (see the next section)

Figure 8 shows the result of running these analyses on lvg-m and resembles Figure 7 where similar analyses where made by tiling. Actually, several of the analyses, such as `da0_t2` and `da0_m2`, and `da0_t6` and `da0_m6`, respectively, have run times that are identical within the accuracy of measurement. For some of the other analyses, the use of memoisation is superior to the tiling transformation. This happens e.g. with the variable t_1 where the analysis using existential quantification (`da0_m5`) is a constant factor better than the corresponding analysis using tiling (`da0_t5`).

[4] The variable x is only included in the environments within the scope of the quantifier. For the clause (2), the scope ends right after the query $P(x, y_1)$ while it extends to the end of the entire clause in (1). Thus, the environments which contain x will be propagated further in (1) that in (2). Still, the exact same *number* of environments would be propagated in two clauses if the Succinct Solver did not have memoisation at existential quantification.

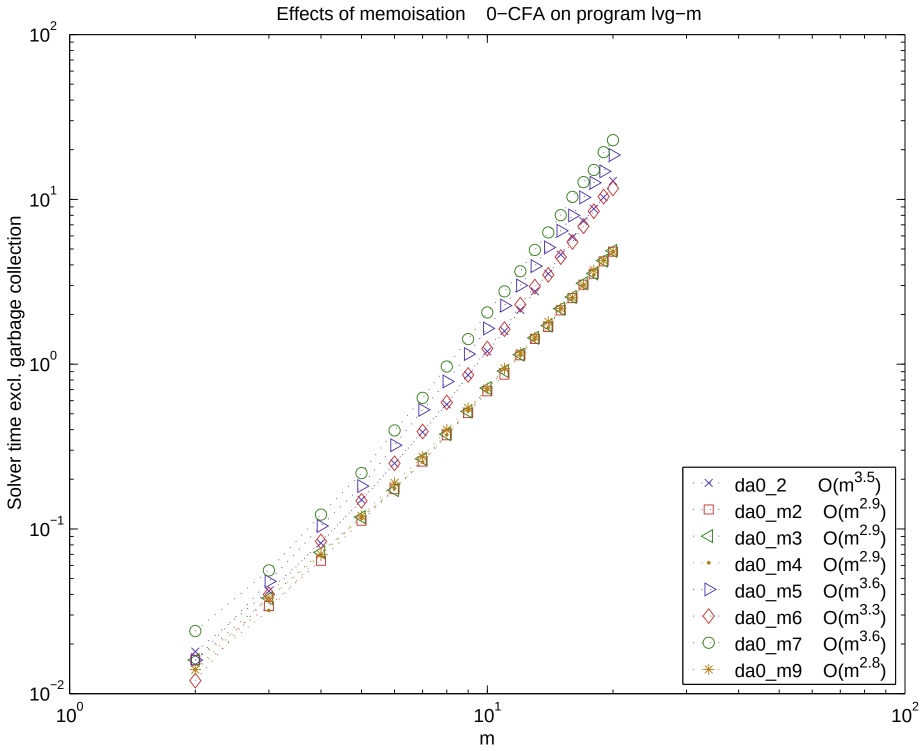


Fig. 8.

Even better is the analysis `da0_m7` over the analysis `da0_t7`. Here the improvement of memoisation over tiling is by a degree in the complexity polynomial. In the analysis `da0_t7` tiling of e.g. the variable μ^p generates a *sibling* relation. This relation becomes very large and the high complexity of this analysis is due to the calculation of this relation. On the other hand, when using memoisation as in `da0_m7` this relation is not build explicitly. Instead it is evaluated only when needed, i.e. in a "lazy" fashion, thereby saving the work of generating the entire relation, which also contains unused elements. Thus, memoisation can be more efficient than the corresponding tiling transformation.

The above results are similar, for almost[5] all the test programs as shown in Appendix XXVII.

[5] The analysis `da0_m7` is only a constant factor better than the analysis `da0_m7` for the program `sph`-m. The reason is the analysis itself has a complexity, which is as bad as the calculation of the sibling relation. Thus, we cannot expect to improve the overall complexity by "optimising the calculation of the sibling relation".

5.3 Greedy Memoisation

The experiments presented in the previous section seem to show that using existential quantifiers is *always* a good idea. We have tested this hypothesis by applying a *greedy* transformation which replaces universal quantifier in clauses with existential quantifiers in the precondition *whenever* it is possible. The resulting analyses are `da0_m9` for 0-CFA (see Appendix XXIII) and `da1_m9` for 1-CFA.

Experiments on the various test programs are displayed on Figure 8 and Appendix XXVII for 0-CFA and in Appendix XXVIII for the 1-CFA. They show that this greedy approach is a good idea. In many of the experiments, the two analyses `da0_m9` and `da1_m9` are the best of all the analysis we have presented in the paper. At times, however, this is not the case but in these cases the two analyses are only small *constant* factors worse than the best of the analyses we have found.

This greedy approach of replacing universal quantifiers with existential quantifiers, seem to have similar performance characteristics as the transformation, which reorders parameters in relations. Since, the greedy transformation is a purely syntactic transformation it, too, can be fully automated. However, as was the case for the reordering transformation also this transformation trades memory in return for the chance of a great increase of the solving time. There is, however, cases where the transformation does not yield a more time efficient clause; so, instead, memory is wasted. Additionally, the greedy memoisation scheme, so far, works only for Horn Clause, therefore, cannot be applied to arbitrary Alternation-free Least Fixed Point formulae.

6 Tuning Clauses

We summarise our results in this section by stating a number of recommendations on how to tune clauses to be efficiently solvable. For an explanation of the rational behind our recommendations the reader is directed to the previous sections.

Ordering parameters in relations. The order in which parameters appear in relations as queries and conclusions is *not* a concern when writing clauses. The Succinct Solver automatically reorders the parameters in an "optimal" way before solving and this does not require user interaction. It may, however, sometimes be possible to optimise a clause for space consumption by close inspection and manual reordering of the parameters as described in section 3.

Ordering conjuncts in preconditions. Preconditions are evaluated from left to right by propagating environments. Thus, the fewer environments that are propagated, the more efficient it is to evaluate the precondition. Queries that bind a variable to *few* possible values should, therefore, appear further to the left in a precondition than queries that bind the variable to many values. This transformation must be performed manually.

Existential quantification in preconditions. Whenever possible, existential quantification in preconditions should be used instead of universal quantification of the entire clause. Thus, a clause on the form a) is preferred over a clause on the form b) below

$$\begin{array}{ll} \text{a)} & \forall y_1,\dots,y_n[\ \exists x\ [\ \mathit{pre}_x\ ] \wedge \mathit{pre}_y\ ] \Rightarrow \mathit{con} \\ \text{b)} & \forall x, y_1,\dots,y_n\ [\ \mathit{pre}_x \wedge \mathit{pre}_y\ ] \Rightarrow \mathit{con} \end{array}$$

where x is a variable that is *not* free in pre_y or con.

Sharing preconditions. Sharing a precondition between multiple conclusions, i.e. the use of conjunct in a conclusion, will improve efficiency by *at most* a constant factor. Thus below, a clause on the form a) will, generally, be more efficient that b).

$$\text{a)}\ \ \mathit{pre} \Rightarrow \mathit{con}_1 \wedge \mathit{con}_2 \qquad \text{b)}\ \ \begin{array}{l} \mathit{pre} \Rightarrow \mathit{con}_1 \wedge \\ \mathit{pre} \Rightarrow \mathit{con}_2 \end{array}$$

However, if the precondition *pre* is small (e.g. it only contains one or two conjuncts) the use of sharing will have no measurable effect and may be disregarded.

7 Conclusion

Through a series of experiments involving 0-CFA and 1-CFA analyses for Discretionary Ambients we have studied how systematic transformations on the input clauses to the Succinct Solver effect solving time. We have commented on the experiments, thus, explaining the coherence between the solving algorithm and empirical observations. Finally, we have suggested a number of transformation the will make clauses more efficiently solvable. The main next step is to use these insights for tuning the analysis developed for Carmel (see SECSAFE-IMM-001).

References

1. A. Aiken. Introduction to set constraint-based program analysis. *Science of Computer Programming (SCP)*, 35(2):79–111, 1999.
2. Baudouin Le Charlier and Pascal Van Hentenryck. A Universal Top-Down Fixpoint Algorithm. Technical Report CS-92-25, Brown University, Providence, RI 02912, 1992.

3. Christian Fecht and Helmut Seidl. Propagating Differences: An Efficient New Fixpoint Algorithm for Distributive Constraint Systems. In *European Symposium on Programming (ESOP)*, pages 90–104. LNCS 1381, Springer Verlag, 1998. Long version in *Nordic Journal of Computing 5, 304-329, 1998.*
4. Christian Fecht and Helmut Seidl. A Faster Solver for General Systems of Equations. *Science of Computer Programming (SCP)*, 35(2-3):137–162, 1999.
5. F. Martin. PAG – an efficient program analyzer generator. *Journal of Software Tools for Technology Transfer*, 2, 1998.
6. David McAllester. On the Complexity Analysis of Static Analyses. In *6th Static Analysis Symposium (SAS)*, pages 312–329. LNCS 1694, Springer Verlag, 1999.
7. F. Nielsen and H. Seidl. Control-flow analysis in cubic time. In *10th European Symposium on Programming (ESOP)*, pages 252–268. LNCS 2028, Springer Verlag, 2001.
8. F. Nielson and H. Seidl. Succinct solvers. Technical Report 01-12, University of Trier, Germany, 2001.
9. H. Riis Nielson and F. Nielson. Discretionary Ambients. Manuscript, 2001.

I 0-CFA – The Initial Specification

$$\mathcal{I} \models^{\star}_{\Gamma} (\nu n : \mu)P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma[n\mapsto\mu]} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} (\nu \mu)P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} \mathbf{0}$$

$$\mathcal{I} \models^{\star}_{\Gamma} P_1 \mid P_2 \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} P_1 \land \mathcal{I} \models^{\star}_{\Gamma} P_2$$

$$\mathcal{I} \models^{\star}_{\Gamma} !P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} n[P] \quad \underline{\mathsf{iff}}\ \mu \in \mathcal{I}(\star) \land \mathcal{I} \models^{\mu}_{\Gamma} P \quad \underline{\mathsf{where}} \quad \mu = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} M.P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} M \land \mathcal{I} \models^{\star}_{\Gamma,\mathcal{L}} P$$

$$\begin{array}{l}
\mathcal{I} \models^{\star}_{\Gamma} \mathtt{in}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{in}(\mu) \in \mathcal{I}(\star)\ \land \\ \forall\, \mu^a, \mu^p : \begin{bmatrix} \mathsf{in}(\mu) \in \mathcal{I}(\mu^a)\ \land \\ \mu^a \in \mathcal{I}(\mu^p)\ \land \\ \mu \in \mathcal{I}(\mu^p)\ \land \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle \in \mathcal{I}(\mu) \end{bmatrix} \Rightarrow \mu^a \in \mathcal{I}(\mu) \end{bmatrix} \\
\qquad\qquad \underline{\mathsf{where}}\ \mu = \Gamma(n)
\end{array}$$

$$\begin{array}{l}
\mathcal{I} \models^{\star}_{\Gamma} \mathsf{out}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{out}(\mu) \in \mathcal{I}(\star)\ \land \\ \forall\, \mu^a, \mu^g : \begin{bmatrix} \mathsf{out}(\mu) \in \mathcal{I}(\mu^a)\ \land \\ \mu^a \in \mathcal{I}(\mu)\ \land \\ \mu \in \mathcal{I}(\mu^g)\ \land \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle \in \mathcal{I}(\mu) \end{bmatrix} \Rightarrow \mu^a \in \mathcal{I}(\mu^g) \end{bmatrix} \\
\qquad\qquad \underline{\mathsf{where}}\ \mu = \Gamma(n)
\end{array}$$

$$\begin{array}{l}
\mathcal{I} \models^{\star}_{\Gamma} \mathsf{open}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{open}(\mu) \in \mathcal{I}(\star)\ \land \\ \forall\, \mu^p : \begin{bmatrix} \mathsf{open}(\mu) \in \mathcal{I}(\mu^p)\ \land \\ \mu \in \mathcal{I}(\mu^p)\ \land \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle \in \mathcal{I}(\mu) \end{bmatrix} \Rightarrow \mathcal{I}(\mu) \subseteq \mathcal{I}(\mu^p) \end{bmatrix} \\
\qquad\qquad \underline{\mathsf{where}}\ \mu = \Gamma(n)
\end{array}$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathtt{in}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \overline{\mathsf{in}}\langle \mu, \mu' \rangle \in \mathcal{I}(\star) \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathsf{out}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \overline{\mathsf{out}}\langle \mu, \mu' \rangle \in \mathcal{I}(\star) \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathsf{open}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \overline{\mathsf{open}}\langle \mu, \mu' \rangle \in \mathcal{I}(\star) \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

II 0-CFA – Representation Function

$$\mathcal{I} \models^{\star}_{\Gamma} (\nu n : \mu)P \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \mathsf{new}\langle n, \mu\rangle\rangle \wedge \mathcal{I} \models^{\star}_{\Gamma[n \mapsto \mu]} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} (\nu \mu)P \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \mathsf{group}(\mu)\rangle \wedge \mathcal{I} \models^{\star}_{\Gamma[\mu \mapsto \ell]} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} \mathbf{0}$$

$$\mathcal{I} \models^{\star}_{\Gamma} P_1 \mid P_2 \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} P_1 \wedge \mathcal{I} \models^{\star}_{\Gamma} P_2$$

$$\mathcal{I} \models^{\star}_{\Gamma} !P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} n[P] \quad \underline{\mathsf{iff}}\ \ \textsc{prg}\langle \star, \mathsf{amb}(\mu)\rangle \wedge \mathcal{I} \models^{\mu}_{\Gamma} P \quad \underline{\mathsf{where}} \quad \mu = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} M.P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models^{\star}_{\Gamma} M \wedge \mathcal{I} \models^{\star}_{\Gamma} P$$

$$\mathcal{I} \models^{\star}_{\Gamma} \mathtt{in}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \mathsf{in}(\mu)\rangle \quad \underline{\mathsf{where}} \quad \mu = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \mathsf{out}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \mathsf{out}(\mu)\rangle \quad \underline{\mathsf{where}} \quad \mu = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \mathsf{open}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \mathsf{open}(\mu)\rangle \quad \underline{\mathsf{where}} \quad \mu = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathtt{in}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \overline{\mathsf{in}}\langle \mu, \mu'\rangle\rangle \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathsf{out}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \overline{\mathsf{out}}\langle \mu, \mu'\rangle\rangle \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

$$\mathcal{I} \models^{\star}_{\Gamma} \overline{\mathsf{open}}_{\mu}\ n \quad \underline{\mathsf{iff}}\ \textsc{prg}\langle \star, \overline{\mathsf{open}}\langle \mu, \mu'\rangle\rangle \quad \underline{\mathsf{where}} \quad \mu' = \Gamma(n)$$

III Closure Condition: `da0_1`

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \; \wedge \\ \forall \mu^a, \mu^p, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^a, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \; \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \; \wedge \\ \forall \mu^a, \mu^g, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^a, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \; \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \; \wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \; \wedge \\ \forall \mu^p, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^p, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \; \wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \end{bmatrix} \Rightarrow \forall u_1 : \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

IV Closure Condition: da0_2

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^g, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{bmatrix} \Rightarrow \forall u_1 : \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

V Closure Condition: da0_3

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall\, \mu^a, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall\, \mu^a, \mu^g, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall\, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \end{bmatrix} \Rightarrow \forall\, u_1 : \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

VI Closure Condition: `da0_4`

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \land \\ \mathsf{HasIn}\langle \mu, t_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \land \\ \mathsf{HasOut}\langle \mu, t_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \star, t_1 \rangle \land \\ \mathsf{HasOpen}\langle \mu, t_1 \rangle \end{bmatrix}$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \land \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \mu^p, \mu : \mathsf{S_p}\langle \mu, \mu^p \rangle \Rightarrow \forall u_1 : \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall t_2, t_1, \mu^p, \mu, \mu^a : \begin{bmatrix} \mathsf{HasIn}\langle \mu, t_1 \rangle \land \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \land \\ \mathcal{I}\langle \mu, t_2 \rangle \land \\ \mathcal{I}\langle \mu^p, \mu \rangle \land \\ \mathcal{I}\langle \mu^a, t_1 \rangle \land \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall t_2, t_1, \mu, \mu^g, \mu^a : \begin{bmatrix} \mathsf{HasOut}\langle \mu, t_1 \rangle \land \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \land \\ \mathcal{I}\langle \mu, t_2 \rangle \land \\ \mathcal{I}\langle \mu^g, \mu \rangle \land \\ \mathcal{I}\langle \mu^a, t_1 \rangle \land \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall t_2, t_1, \mu, \mu^p : \begin{bmatrix} \mathsf{HasOpen}\langle \mu, t_1 \rangle \land \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \land \\ \mathcal{I}\langle \mu, t_2 \rangle \land \\ \mathcal{I}\langle \mu^p, \mu \rangle \land \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{bmatrix} \Rightarrow \mathsf{S_p}\langle \mu, \mu^p \rangle$$

VII Closure Condition: da0_6fifi

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^p, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right]$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^g, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{out}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle \end{array}\right]$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^p, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{open}}\langle \mu, \mu^p \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{array}\right] \Rightarrow \forall u_1 : \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle \end{array}\right]$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu', \mu \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu', \mu \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu', \mu \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

VIII Closure Condition: da0_7

$$\forall \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \mathsf{amb}(\mu) = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \mathsf{in}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^p, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right]$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \mathsf{out}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^a, \mu^g, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{out}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle \end{array}\right]$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \mathsf{open}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \star, t_1 \rangle \wedge \\ \forall \mu^p, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \overline{\mathsf{open}}\langle \mu, \mu^p \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{array}\right] \Rightarrow \forall u_1\ \mathcal{I}\langle \mu, u_1 \rangle \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle \end{array}\right]$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu', \mu \rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \overline{\mathsf{out}}\langle \mu', \mu \rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle t_1, \star \rangle \wedge \\ \overline{\mathsf{open}}\langle \mu', \mu \rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

IX 1-CFA – The Initial Specification

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} (\nu n : \mu)P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models_{\Gamma[n\mapsto\mu]}^{\langle\top,\star\rangle} P$$

$$\underline{\mathsf{iff}}\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} (\nu\mu)P\ell$$

$$\mathcal{I} \models_{\Gamma[\mu\mapsto\ell]}^{\langle\top,\star\rangle} P \quad \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathbf{0}$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_1 \mid P_2 \quad \underline{\mathsf{iff}} \begin{bmatrix} \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_1 \ \wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_2 \end{bmatrix}$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} !P \quad \underline{\mathsf{iff}}\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} n[P] \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mu \in \mathcal{I}\langle\top,\star\rangle \ \wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\star,\mu\rangle} P \end{bmatrix}$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} M.P \quad \underline{\mathsf{iff}} \begin{bmatrix} \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} M \ \wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P \end{bmatrix}$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathtt{in}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{in}(\mu) \in \mathcal{I}\langle\top,\star\rangle \ \wedge \\ \forall\, \mu^a, \mu^p, \mu^q : \begin{bmatrix} \mathsf{in}(\mu) \in \mathcal{I}\langle\mu^p,\mu^a\rangle \ \wedge \\ \mu^a \in \mathcal{I}\langle\mu^q,\mu^p\rangle \ \wedge \\ \mu \in \mathcal{I}\langle\mu^q,\mu^p\rangle \ \wedge \\ \overline{\mathsf{in}}\langle\mu^a,\mu\rangle \in \mathcal{I}\langle\mu^p,\mu\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mu^a \in \mathcal{I}\langle\mu^p,\mu\rangle \ \wedge \\ \mathcal{I}\langle\mu^p,\mu^a\rangle \subseteq \mathcal{I}\langle\mu,\mu^a\rangle \end{bmatrix} \end{bmatrix}$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathtt{out}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{out}(\mu) \in \mathcal{I}\langle\top,\star\rangle \ \wedge \\ \forall\, \mu^a, \mu^g, \mu^q : \begin{bmatrix} \mathsf{out}(\mu) \in \mathcal{I}\langle\mu,\mu^a\rangle \ \wedge \\ \mu^a \in \mathcal{I}\langle\mu^g,\mu\rangle \ \wedge \\ \mu \in \mathcal{I}\langle\mu^q,\mu^g\rangle \ \wedge \\ \overline{\mathsf{out}}\langle\mu^a,\mu\rangle \in \mathcal{I}\langle\mu^g,\mu\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mu^a \in \mathcal{I}\langle\mu^q,\mu^g\rangle \ \wedge \\ \mathcal{I}\langle\mu,\mu^a\rangle \subseteq \mathcal{I}\langle\mu^g,\mu^a\rangle \end{bmatrix} \end{bmatrix}$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathtt{open}\ n \quad \underline{\mathsf{iff}} \quad \begin{bmatrix} \mathsf{open}(\mu) \in \mathcal{I}\langle\top,\star\rangle \ \wedge \\ \forall\, \mu^p, \mu^q : \begin{bmatrix} \mathsf{open}(\mu) \in \mathcal{I}\langle\mu^q,\mu^p\rangle \ \wedge \\ \mu \in \mathcal{I}\langle\mu^q,\mu^p\rangle \ \wedge \\ \overline{\mathsf{open}}\langle\mu^p,\mu\rangle \in \mathcal{I}\langle\mu^p,\mu\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle\mu^p,\mu\rangle \subseteq \mathcal{I}\langle\mu^q,\mu^p\rangle \end{bmatrix} \end{bmatrix}$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\mathtt{in}}_{\mu}\ n \quad \underline{\mathsf{iff}} \quad \overline{\mathsf{in}}\langle\mu,\mu'\rangle \in \mathcal{I}\langle\top,\star\rangle$$
$$\underline{\mathsf{where}}\ \mu' = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\texttt{out}}_{\mu}\ n \quad \underline{\textsf{iff}} \quad \overline{\textsf{out}}\langle\mu,\mu'\rangle \in \mathcal{I}\langle\top,\star\rangle$$
$$\underline{\textsf{where}}\ \mu' = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\texttt{open}}_{\mu}\ n \quad \underline{\textsf{iff}} \quad \overline{\textsf{open}}\langle\mu,\mu'\rangle \in \mathcal{I}\langle\top,\star\rangle$$
$$\underline{\textsf{where}}\ \mu' = \Gamma(n)$$

X 1-CFA – Representation Function

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} (\nu n : \mu)P \quad \underline{\mathsf{iff}} \left[\begin{array}{l} \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{new}\langle n,\mu\rangle\rangle \;\wedge \\ \mathcal{I} \models_{\Gamma[n\mapsto\mu]}^{\langle\top,\star\rangle} P \end{array}\right]$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} (\nu\mu)P\ell \quad \underline{\mathsf{iff}} \left[\begin{array}{l} \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{group}\langle\mu,\ell\rangle\rangle \;\wedge \\ \mathcal{I} \models_{\Gamma[\mu\mapsto\ell]}^{\langle\top,\star\rangle} P \end{array}\right]$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathbf{0}$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_1 \mid P_2 \quad \underline{\mathsf{iff}} \left[\begin{array}{l} \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_1 \;\wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P_2 \end{array}\right]$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} !P \quad \underline{\mathsf{iff}}\, \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} n[P] \quad \underline{\mathsf{iff}} \quad \left[\begin{array}{l} \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{amb}(\mu)\rangle \;\wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\star,\mu\rangle} P \end{array}\right]$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} M.P \quad \underline{\mathsf{iff}} \left[\begin{array}{l} \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} M \;\wedge \\ \mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} P \end{array}\right]$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathtt{in}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{in}(\mu)\rangle$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathsf{out}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{out}(\mu)\rangle$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \mathsf{open}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \mathsf{open}(\mu)\rangle$$
$$\underline{\mathsf{where}}\ \mu = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\mathtt{in}}_{\mu}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \overline{\mathsf{in}}\langle\mu,\mu'\rangle\rangle$$
$$\underline{\mathsf{where}}\ \mu' = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\mathsf{out}}_{\mu}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \overline{\mathsf{out}}\langle\mu,\mu'\rangle\rangle$$
$$\underline{\mathsf{where}}\ \mu' = \Gamma(n)$$

$$\mathcal{I} \models_{\Gamma}^{\langle\top,\star\rangle} \overline{\mathsf{open}}_{\mu}\ n \quad \underline{\mathsf{iff}} \quad \textsc{prg}\langle\langle\top,\star\rangle, \overline{\mathsf{open}}\langle\mu,\mu'\rangle\rangle$$
$$\underline{\mathsf{where}}\ \mu' = \Gamma(n)$$

XI Closure Condition: da1_1

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, \mu\rangle$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^a, \mu^p, \mu^q, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu^p, \mu^a, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu^a\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \end{array}\right] \Rightarrow \\ \qquad\qquad\qquad\qquad \left[\begin{array}{l} \mathcal{I}\langle \mu^p, \mu, \mu^a\rangle \wedge \\ \forall u_1 : \mathcal{I}\langle \mu^p, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu, \mu^a, u_1\rangle \end{array}\right] \end{array}\right]$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^a, \mu^g, \mu^q, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu, \mu^a, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu, \mu^a\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^g, \mu\rangle \wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^g, \mu, t_2\rangle \end{array}\right] \Rightarrow \\ \qquad\qquad\qquad\qquad \left[\begin{array}{l} \mathcal{I}\langle \mu^q, \mu^g, \mu^a\rangle \wedge \\ \forall u_1 : \mathcal{I}\langle \mu, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^g, \mu^a, u_1\rangle \end{array}\right] \end{array}\right]$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^p, \mu^q, t_2 : \left[\begin{array}{l} \mathcal{I}\langle \mu^q, \mu^p, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \end{array}\right] \Rightarrow \\ \qquad\qquad\qquad\qquad \forall u_1 : \mathcal{I}\langle \mu^p, \mu, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^q, \mu^p, u_1\rangle \end{array}\right]$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

XII Closure Condition: da1_2

$$\forall \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, \mu\rangle$$

$$\forall \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^a, \mu^p, \mu^q, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu^a, t_1\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu^a\rangle \end{bmatrix} \Rightarrow \\ \begin{bmatrix} \mathcal{I}\langle \mu^p, \mu, \mu^a\rangle \ \wedge \\ \forall\, u_1 : \mathcal{I}\langle \mu^p, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu, \mu^a, u_1\rangle \end{bmatrix} \end{bmatrix}$$

$$\forall \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^a, \mu^g, \mu^q, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^g, \mu, t_2\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^g, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu, \mu^a, t_1\rangle \ \wedge \\ \mathcal{I}\langle \mu^g, \mu, \mu^a\rangle \end{bmatrix} \Rightarrow \\ \begin{bmatrix} \mathcal{I}\langle \mu^q, \mu^g, \mu^a\rangle \ \wedge \\ \forall\, u_1 : \mathcal{I}\langle \mu, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^g, \mu^a, u_1\rangle \end{bmatrix} \end{bmatrix}$$

$$\forall \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^p, \mu^q, t_2 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, t_1\rangle \end{bmatrix} \Rightarrow \\ \forall\, u_1 : \mathcal{I}\langle \mu^p, \mu, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^q, \mu^p, u_1\rangle \end{bmatrix}$$

$$\forall \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu'\rangle = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu'\rangle = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu'\rangle = t_1 \ \wedge \\ \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

XIII Closure Condition: `da1_3`

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{amb}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, \mu\rangle$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^a, \mu^p, \mu^q, t_2 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu^a\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \end{array}\right] \Rightarrow \\ \qquad \left[\begin{array}{l} \mathcal{I}\langle \mu^p, \mu, \mu^a\rangle \wedge \\ \forall u_1 : \mathcal{I}\langle \mu^p, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu, \mu^a, u_1\rangle \end{array}\right] \end{array}\right]$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^a, \mu^g, \mu^q, t_2 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^g, \mu, t_2\rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu, \mu^a\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^g, \mu\rangle \end{array}\right] \Rightarrow \\ \qquad \left[\begin{array}{l} \mathcal{I}\langle \mu^q, \mu^g, \mu^a\rangle \wedge \\ \forall u_1 : \mathcal{I}\langle \mu, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^g, \mu^a, u_1\rangle \end{array}\right] \end{array}\right]$$

$$\forall \top, \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \wedge \\ \forall \mu^p, \mu^q, t_2 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu^p, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu^p, \mu, t_2\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, t_1\rangle \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \end{array}\right] \Rightarrow \\ \qquad \forall u_1 : \mathcal{I}\langle \mu^p, \mu, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^q, \mu^p, u_1\rangle \end{array}\right]$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall \top, \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu, \mu'\rangle = t_1 \wedge \\ \mathrm{PRG}\langle\langle \top, \star\rangle, t_1\rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

XIV Closure Condition: `da1_6gfiigf`

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \mathsf{lifg}\langle \mu, \star, \top \rangle$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_1, \star, \top \rangle \wedge \\ \forall\, \mu^a, \mu^p, \mu^q, t_2 : \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_2, \mu, \mu^p \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathsf{lifg}\langle \mu, \mu^p, \mu^q \rangle \wedge \\ \mathsf{lifg}\langle t_1, \mu^a, \mu^p \rangle \wedge \\ \mathsf{lifg}\langle \mu^a, \mu^p, \mu^q \rangle \end{array} \end{bmatrix} \Rightarrow \\ \qquad \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle \mu^a, \mu, \mu^p \rangle \wedge \\ \forall\, u_1 : \mathsf{lifg}\langle u_1, \mu^a, \mu^p \rangle \Rightarrow \mathsf{lifg}\langle u_1, \mu^a, \mu \rangle \end{array} \end{bmatrix} \end{array} \end{bmatrix}$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_1, \star, \top \rangle \wedge \\ \forall\, \mu^a, \mu^g, \mu^q, t_2 : \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_2, \mu, \mu^g \rangle \wedge \\ \overline{\mathsf{out}}\langle \mu, \mu^a \rangle = t_2 \wedge \\ \mathsf{lifg}\langle \mu, \mu^g, \mu^q \rangle \wedge \\ \mathsf{lifg}\langle t_1, \mu^a, \mu \rangle \wedge \\ \mathsf{lifg}\langle \mu^a, \mu, \mu^g \rangle \end{array} \end{bmatrix} \Rightarrow \\ \qquad \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle \mu^a, \mu^g, \mu^q \rangle \wedge \\ \forall\, u_1 : \mathsf{lifg}\langle u_1, \mu^a, \mu \rangle \Rightarrow \mathsf{lifg}\langle u_1, \mu^a, \mu^g \rangle \end{array} \end{bmatrix} \end{array} \end{bmatrix}$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_1, \star, \top \rangle \wedge \\ \forall\, \mu^p, \mu^q, t_2 : \begin{bmatrix} \begin{array}{l} \mathsf{lifg}\langle t_2, \mu, \mu^p \rangle \wedge \\ \overline{\mathsf{open}}\langle \mu, \mu^p \rangle = t_2 \wedge \\ \mathsf{lifg}\langle \mu, \mu^p, \mu^q \rangle \wedge \\ \mathsf{lifg}\langle t_1, \mu^p, \mu^q \rangle \end{array} \end{bmatrix} \Rightarrow \\ \qquad \forall\, u_1 : \mathsf{lifg}\langle u_1, \mu, \mu^p \rangle \Rightarrow \mathsf{lifg}\langle u_1, \mu^p, \mu^q \rangle \end{array} \end{bmatrix}$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu', \mu \rangle = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \mathsf{lifg}\langle t_1, \star, \top \rangle$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu', \mu \rangle = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \mathsf{lifg}\langle t_1, \star, \top \rangle$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu', \mu \rangle = t_1 \wedge \\ \mathrm{PRG}\langle t_1, \langle \top, \star \rangle\rangle \end{bmatrix} \Rightarrow \mathsf{lifg}\langle t_1, \star, \top \rangle$$

XV Closure Condition: `da1_7`

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \mathsf{amb}(\mu) = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, \mu\rangle$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \mathsf{in}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^a, \mu^p, \mu^q, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^p, \mu, t_2\rangle \ \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu^a, t_1\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu^a\rangle \end{bmatrix} \Rightarrow \\ \qquad\qquad\qquad\qquad \begin{bmatrix} \mathcal{I}\langle \mu^p, \mu, \mu^a\rangle \ \wedge \\ \forall\, u_1 : \mathcal{I}\langle \mu^p, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu, \mu^a, u_1\rangle \end{bmatrix} \end{array}\right]$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \mathsf{out}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^a, \mu^g, \mu^q, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^g, \mu, t_2\rangle \ \wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^g, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu, \mu^a, t_1\rangle \ \wedge \\ \mathcal{I}\langle \mu^g, \mu, \mu^a\rangle \end{bmatrix} \Rightarrow \\ \qquad\qquad\qquad\qquad \begin{bmatrix} \mathcal{I}\langle \mu^q, \mu^g, \mu^a\rangle \ \wedge \\ \forall\, u_1 : \mathcal{I}\langle \mu, \mu^a, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^g, \mu^a, u_1\rangle \end{bmatrix} \end{array}\right]$$

$$\forall\, \top, \star, \mu, t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \mathsf{open}(\mu) = t_1 \end{bmatrix} \Rightarrow \left[\begin{array}{l} \mathcal{I}\langle \top, \star, t_1\rangle \ \wedge \\ \forall\, \mu^p, \mu^q, t_2 : \begin{bmatrix} \mathcal{I}\langle \mu^p, \mu, t_2\rangle \ \wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu\rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, \mu\rangle \ \wedge \\ \mathcal{I}\langle \mu^q, \mu^p, t_1\rangle \end{bmatrix} \Rightarrow \\ \qquad\qquad\qquad\qquad \forall\, u_1 : \mathcal{I}\langle \mu^p, \mu, u_1\rangle \Rightarrow \mathcal{I}\langle \mu^q, \mu^p, u_1\rangle \end{array}\right]$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \overline{\mathsf{in}}\langle \mu, \mu'\rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \overline{\mathsf{out}}\langle \mu, \mu'\rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

$$\forall\, \top, \star, \mu, \mu', t_1 : \begin{bmatrix} \textsc{prg}\langle\langle \top, \star\rangle, t_1\rangle \ \wedge \\ \overline{\mathsf{open}}\langle \mu, \mu'\rangle = t_1 \end{bmatrix} \Rightarrow \mathcal{I}\langle \top, \star, t_1\rangle$$

XVI Closure Condition: `da0_t1`

Horn Clause form of the analysis `da0_2` (see Appendix IV).

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, t_1, \mu^a, \mu^p, t_2 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \; \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \; \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, t_1, \mu^a, \mu^g, t_2 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \; \wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \; \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \; \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, t_1, \mu^p, t_2, u_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \; \wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \; \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \; \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \; \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \; \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\textsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\textsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

XVII Closure Condition: `da0_t3`

$\star$ then t_1 tiled from `da0_t1`.

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \mu, \mu^a, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \ \wedge \\ \mathsf{HasIn}\langle \mu, \mu^a \rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \mu, \mu^a, \mu^g, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \ \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \ \wedge \\ \mathsf{HasOut}\langle \mu, \mu^a \rangle \ \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \mu, \mu^p, t_2, u_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \ \wedge \\ \mathsf{HasOpen}\langle \mu, \mu^p \rangle \ \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall\,\star, t\ {:}\textsc{prg}\langle\star, t\rangle \Rightarrow \mathsf{Has}(t)$$

$$\forall\, t_1, \mu, \mu^a : \begin{bmatrix} \mathsf{in}(\mu) = t_1\ \wedge \\ \mathsf{Has}(t_1)\ \wedge \\ \mathcal{I}\langle\mu^a, t_1\rangle \end{bmatrix} \Rightarrow \mathsf{HasIn}\langle\mu, \mu^a\rangle$$

$$\forall\, t_1, \mu, \mu^a : \begin{bmatrix} \mathsf{out}(\mu) = t_1\ \wedge \\ \mathsf{Has}(t_1)\ \wedge \\ \mathcal{I}\langle\mu^a, t_1\rangle \end{bmatrix} \Rightarrow \mathsf{HasOut}\langle\mu, \mu^a\rangle$$

$$\forall\, t_1, \mu, \mu^a : \begin{bmatrix} \mathsf{open}(\mu) = t_1\ \wedge \\ \mathsf{Has}(t_1)\ \wedge \\ \mathcal{I}\langle\mu^a, t_1\rangle \end{bmatrix} \Rightarrow \mathsf{HasOpen}\langle\mu, \mu^a\rangle$$

XVIII Closure Condition: `da0_t5`

t_1 tiled from `da0_t1`.

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^a, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathsf{HasIn}\langle \star, \mu, \mu^a \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^a, \mu^g, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathsf{HasOut}\langle \star, \mu, \mu^a \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^p, t_2, u_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathsf{HasOpen}\langle \star, \mu, \mu^p \rangle \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \mathrm{PRG}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall\, t_1, \star, \mu, \mu^a : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{bmatrix} \Rightarrow \mathsf{HasIn}\langle \star, \mu, \mu^a \rangle$$

$$\forall\, t_1, \star, \mu, \mu^a : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{bmatrix} \Rightarrow \mathsf{HasOut}\langle \star, \mu, \mu^a \rangle$$

$$\forall\, t_1, \star, \mu, \mu^a : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{bmatrix} \Rightarrow \mathsf{HasOpen}\langle \star, \mu, \mu^a \rangle$$

XIX Closure Condition: `da0_t7`

μ^p tiled in in clause and μ tiled in out, and open clause.

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, t_1, \mu^a, t_2 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \ \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \ \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \ \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \ \wedge \\ \mathsf{sib}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, t_1, \mu^a, \mu^g, t_2 : \begin{bmatrix} \textsc{prg}\langle \star, t_1 \rangle \ \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \ \wedge \\ \mathsf{outRedex}\langle t_1, t_2, \mu^g, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, t_1, \mu^p, t_2, u_1 : \begin{bmatrix} \textsc{prg}\langle \star, t_1 \rangle \ \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \ \wedge \\ \mathsf{openRedex}\langle t_1, t_2, \mu^p, u_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \ \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \mu^p, \mu, \mu^a : \begin{bmatrix} \mathcal{I}\langle \mu^p, \mu \rangle \ \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathsf{sib}\langle \mu, \mu^a \rangle$$

$$\forall\, \mu, t_1, t_2, \mu^g, \mu^a : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \;\wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \;\wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \;\wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \;\wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathsf{outRedex}\langle t_1, t_2, \mu^g, \mu^a \rangle$$

$$\forall\, \mu, t_1, t_2, \mu^p, u_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \;\wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \;\wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \;\wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \;\wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{bmatrix} \Rightarrow \mathsf{openRedex}\langle t_1, t_2, \mu^p, u_1 \rangle$$

XX Closure Condition: `da0_m3`

$\star$ and t_1 existentially quantified.

$$\forall \star, \mu : (\exists\, t_1 : \begin{bmatrix} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix}) \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall\, \mu, \mu^a, \mu^p, t_2 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ (\exists \star, t_1 : \begin{bmatrix} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{bmatrix}) \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall\, \mu, \mu^a, \mu^g, t_2 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ (\exists \star, t_1 : \begin{bmatrix} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{bmatrix}) \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall\, \mu, \mu^p, t_2, u_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ (\exists \star, t_1 : \begin{bmatrix} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{bmatrix}) \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \begin{bmatrix} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{bmatrix} \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

XXI Closure Condition: `da0_m5`

t_1 existentially quantified.

$$\forall \star, \mu : (\exists t_1 : \left[\begin{array}{l} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right]) \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^a, \mu^p, t_2 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ (\exists t_1 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{array}\right]) \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^a, \mu^g, t_2 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ (\exists t_1 : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \end{array}\right]) \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu^p, t_2, u_1 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ (\exists t_1 : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \end{array}\right]) \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, \mu', t_1 : \left[\begin{array}{l} \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

XXII Closure Condition: `da0_m7`

μ^p existentially quantified in in clause and μ existentially quantified in out, and open clause.

$$\forall \star, \mu, t_1 : \left[\begin{array}{l} \mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, \mu \rangle$$

$$\forall \star, t_1 : \left[\begin{array}{l} (\exists\, \mu : \mathsf{in}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu, t_1, \mu^a, t_2 : \left[\begin{array}{l} \mathsf{in}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \overline{\mathsf{in}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ (\exists\, \mu^p : \left[\begin{array}{l} \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a \rangle \end{array}\right]) \end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a \rangle$$

$$\forall \star, t_1 : \left[\begin{array}{l} (\exists\, \mu : \mathsf{out}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, t_1, \mu^a, \mu^g, t_2 : \left[\begin{array}{l} \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^a, t_1 \rangle \wedge \\ (\exists\, \mu : \left[\begin{array}{l} \mathsf{out}(\mu) = t_1 \wedge \\ \overline{\mathsf{out}}\langle \mu^a, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^g, \mu \rangle \wedge \\ \mathcal{I}\langle \mu, \mu^a \rangle \end{array}\right]) \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^g, \mu^a \rangle$$

$$\forall \star, t_1 : \left[\begin{array}{l} (\exists\, \mu : \mathsf{open}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, t_1, \mu^p, t_2, u_1 : \left[\begin{array}{l} \textsc{prg}\langle \star, t_1 \rangle \wedge \\ \mathcal{I}\langle \mu^p, t_1 \rangle \wedge \\ (\exists\, \mu : \left[\begin{array}{l} \mathsf{open}(\mu) = t_1 \wedge \\ \overline{\mathsf{open}}\langle \mu^p, \mu \rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2 \rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu \rangle \wedge \\ \mathcal{I}\langle \mu, u_1 \rangle \end{array}\right]) \end{array}\right] \Rightarrow \mathcal{I}\langle \mu^p, u_1 \rangle$$

$$\forall \star, \mu', t_1 : \left[\begin{array}{l} (\exists\, \mu : \overline{\mathsf{in}}\langle \mu, \mu' \rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu', t_1 : \left[\begin{array}{l} (\exists\, \mu : \overline{\mathsf{out}}\langle \mu, \mu' \rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

$$\forall \star, \mu', t_1 : \left[\begin{array}{l} (\exists\, \mu : \overline{\mathsf{open}}\langle \mu, \mu' \rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1 \rangle \end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1 \rangle$$

XXIII Closure Condition: `da0_m9`

Greedy memoisation, i.e. every possible variable is existentially quantified.

$$\forall \star, \mu :(\exists\, t_1 : \left[\begin{array}{l}\mathsf{amb}(\mu) = t_1 \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right]) \Rightarrow \mathcal{I}\langle \star, \mu\rangle$$

$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu : \mathsf{in}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

$$\forall \mu, \mu^a : \left[\begin{array}{l}(\exists\, t_2 : \left[\begin{array}{l}\overline{\mathsf{in}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2\rangle\end{array}\right]) \wedge \\ (\exists\, t_1 : \left[\begin{array}{l}\mathsf{in}(\mu) = t_1 \wedge \\ (\exists \star : \textsc{prg}\langle \star, t_1\rangle) \wedge \\ \mathcal{I}\langle \mu^a, t_1\rangle\end{array}\right]) \wedge \\ (\exists\, \mu^p : \left[\begin{array}{l}\mathcal{I}\langle \mu^p, \mu\rangle \wedge \\ \mathcal{I}\langle \mu^p, \mu^a\rangle\end{array}\right])\end{array}\right] \Rightarrow \mathcal{I}\langle \mu, \mu^a\rangle$$

$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu : \mathsf{out}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

$$\forall \mu^a, \mu^g :(\exists\, \mu : \left[\begin{array}{l}(\exists\, t_2 : \left[\begin{array}{l}\overline{\mathsf{out}}\langle \mu^a, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2\rangle\end{array}\right]) \wedge \\ \mathcal{I}\langle \mu^g, \mu\rangle \wedge \\ (\exists\, t_1 : \left[\begin{array}{l}\mathsf{out}(\mu) = t_1 \wedge \\ (\exists \star : \textsc{prg}\langle \star, t_1\rangle) \wedge \\ \mathcal{I}\langle \mu^a, t_1\rangle\end{array}\right]) \wedge \\ \mathcal{I}\langle \mu, \mu^a\rangle\end{array}\right]) \Rightarrow \mathcal{I}\langle \mu^g, \mu^a\rangle$$

$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu : \mathsf{open}(\mu) = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

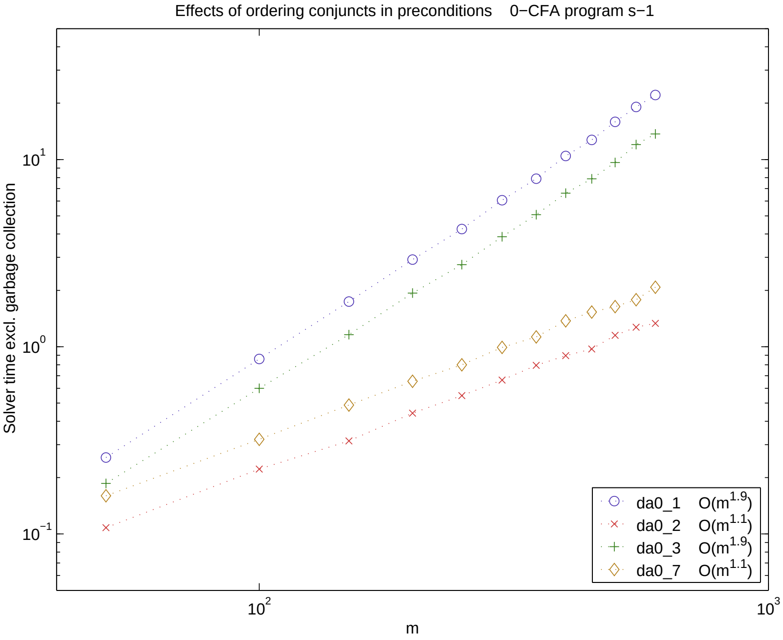
$$\forall \mu^p, u_1 :(\exists\, \mu : \left[\begin{array}{l}(\exists\, t_2 : \left[\begin{array}{l}\overline{\mathsf{open}}\langle \mu^p, \mu\rangle = t_2 \wedge \\ \mathcal{I}\langle \mu, t_2\rangle\end{array}\right]) \wedge \\ \mathcal{I}\langle \mu^p, \mu\rangle \wedge \\ (\exists\, t_1 : \left[\begin{array}{l}\mathsf{open}(\mu) = t_1 \wedge \\ (\exists \star : \textsc{prg}\langle \star, t_1\rangle) \wedge \\ \mathcal{I}\langle \mu^p, t_1\rangle\end{array}\right]) \wedge \\ \mathcal{I}\langle \mu, u_1\rangle\end{array}\right]) \Rightarrow \mathcal{I}\langle \mu^p, u_1\rangle$$

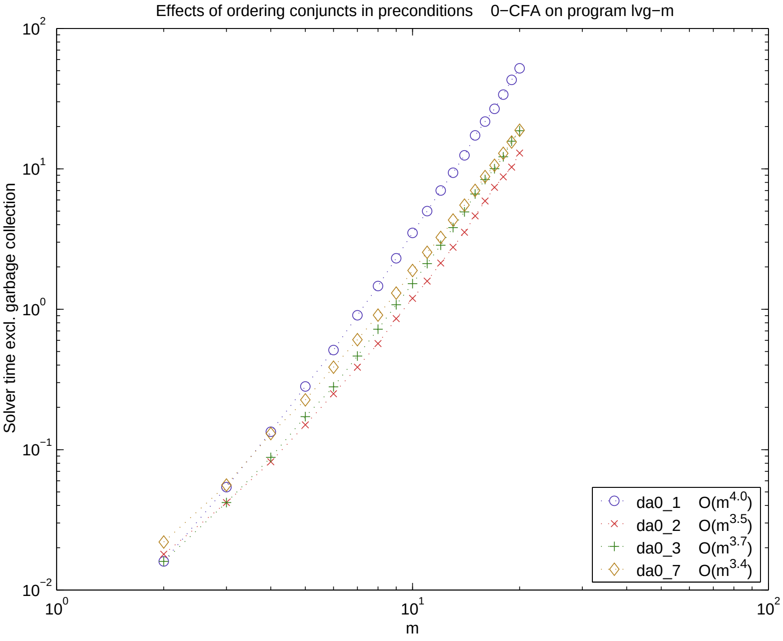
$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu, \mu' : \overline{\mathsf{in}}\langle \mu, \mu'\rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

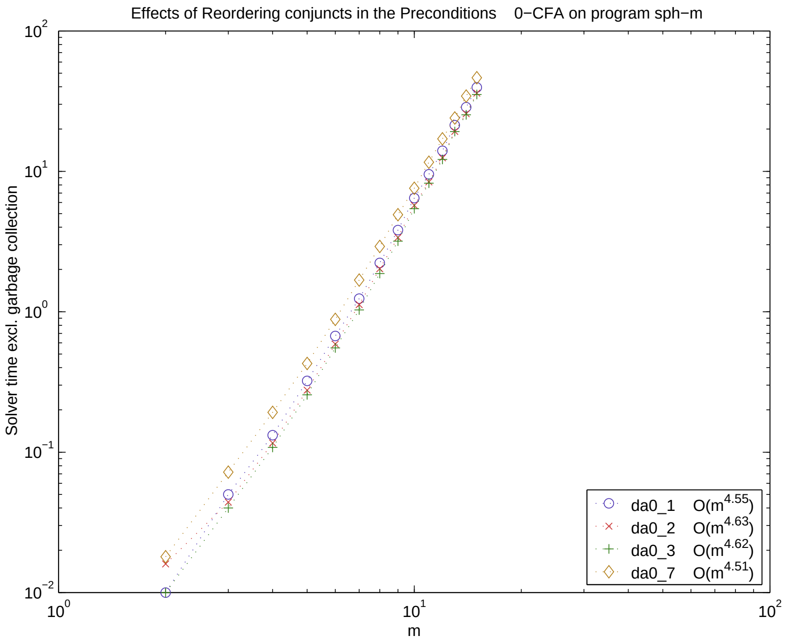
$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu, \mu' : \overline{\mathsf{out}}\langle \mu, \mu'\rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

$$\forall \star, t_1 :\left[\begin{array}{l}(\exists\, \mu, \mu' : \overline{\mathsf{open}}\langle \mu, \mu'\rangle = t_1) \wedge \\ \textsc{prg}\langle \star, t_1\rangle\end{array}\right] \Rightarrow \mathcal{I}\langle \star, t_1\rangle$$

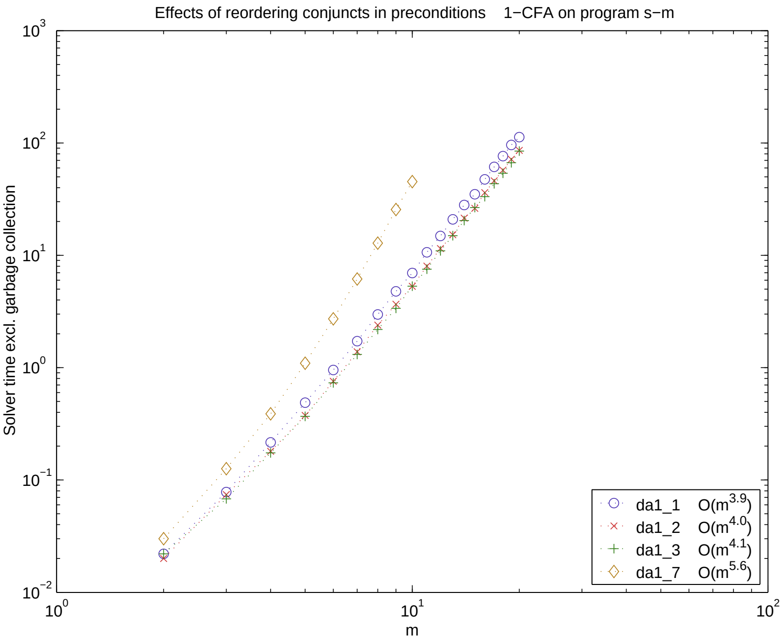
XXIV 0-CFA – Effects of Ordering of Conjuncts in Preconditions

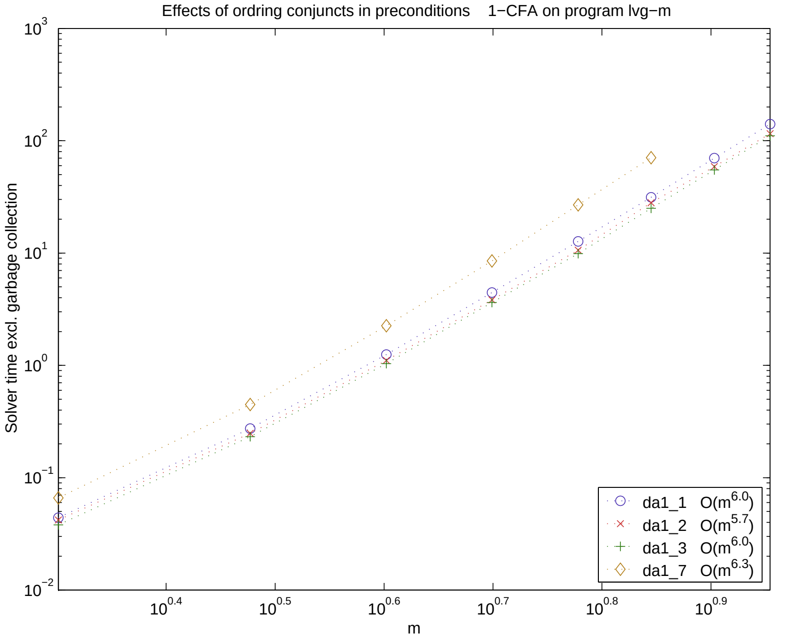


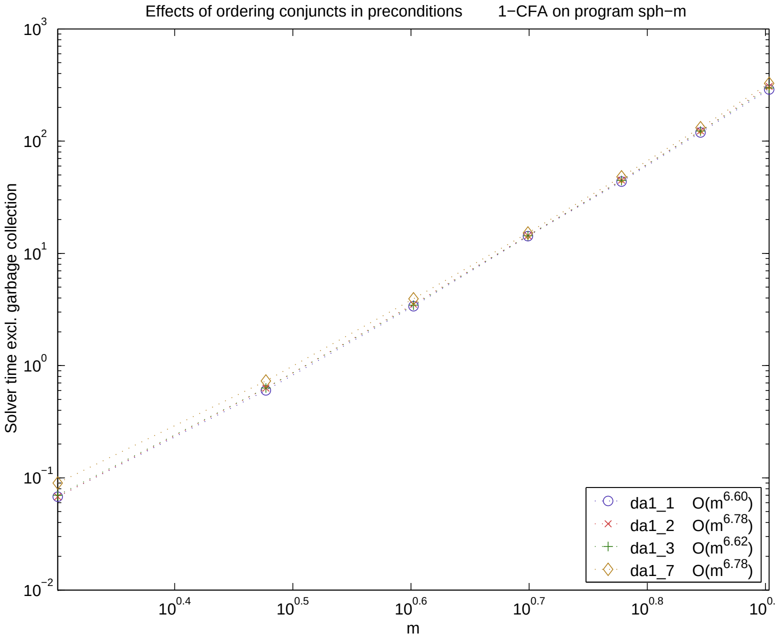



Effects of Reordering conjuncts in the Preconditions   0−CFA on program sph−m
Solver time excl. garbage collection
m
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{0}
10^{1}
10^{2}
○ da0_1 O($m^{4.55}$)
× da0_2 O($m^{4.63}$)
+ da0_3 O($m^{4.62}$)
◇ da0_7 O($m^{4.51}$)

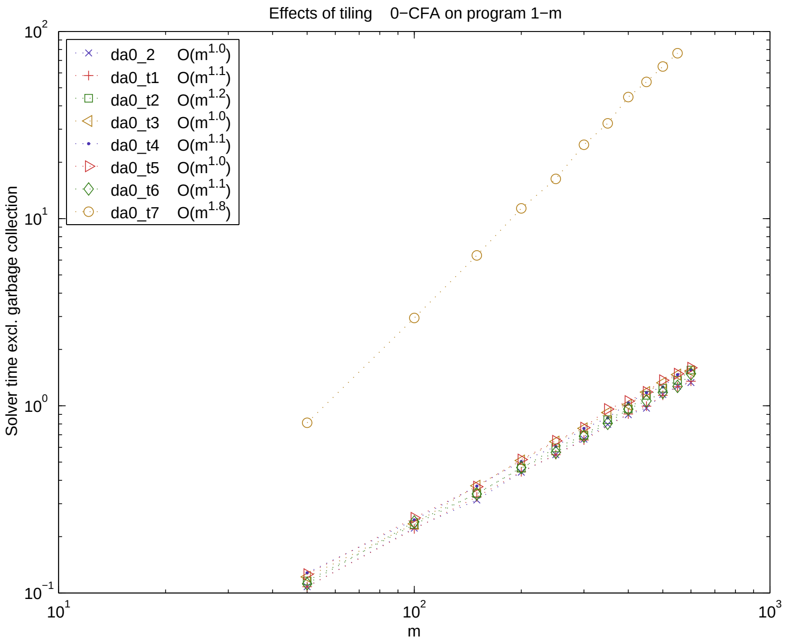
XXV 1-CFA – Effects of Ordering of Conjuncts in Preconditions

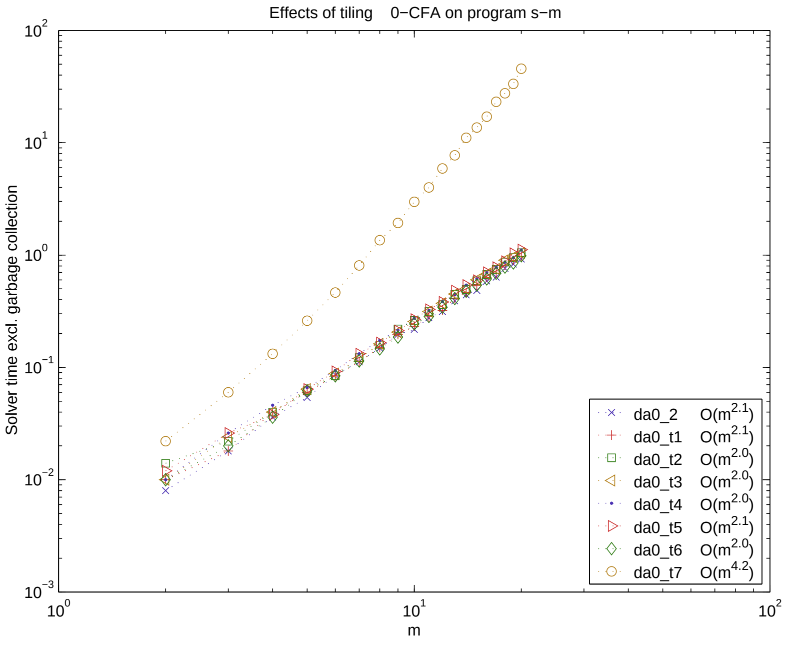


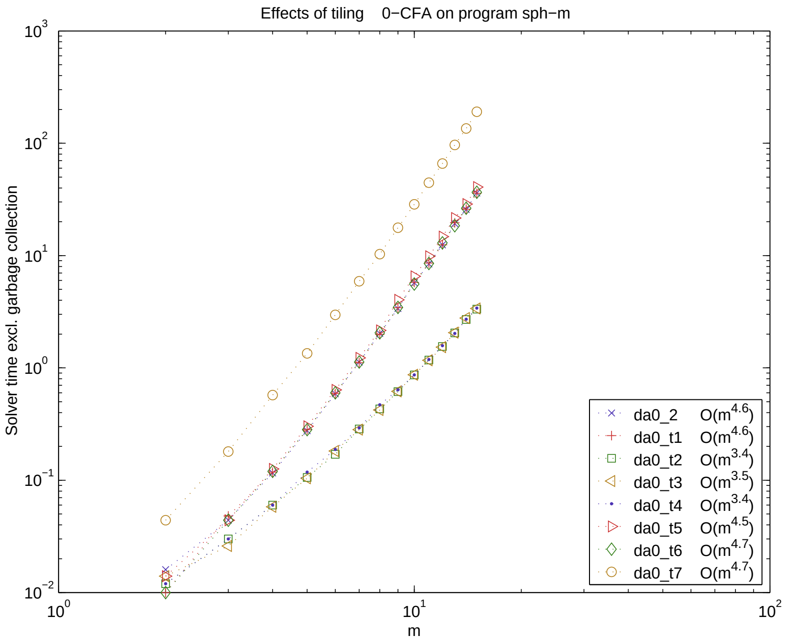



Effects of ordering conjuncts in preconditions 1−CFA on program sph−m
Solver time excl. garbage collection
m
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
$10^{0.4}$
$10^{0.5}$
$10^{0.6}$
$10^{0.7}$
$10^{0.8}$
$10^{0.9}$
da1_1 O($m^{6.60}$)
da1_2 O($m^{6.78}$)
da1_3 O($m^{6.62}$)
da1_7 O($m^{6.78}$)

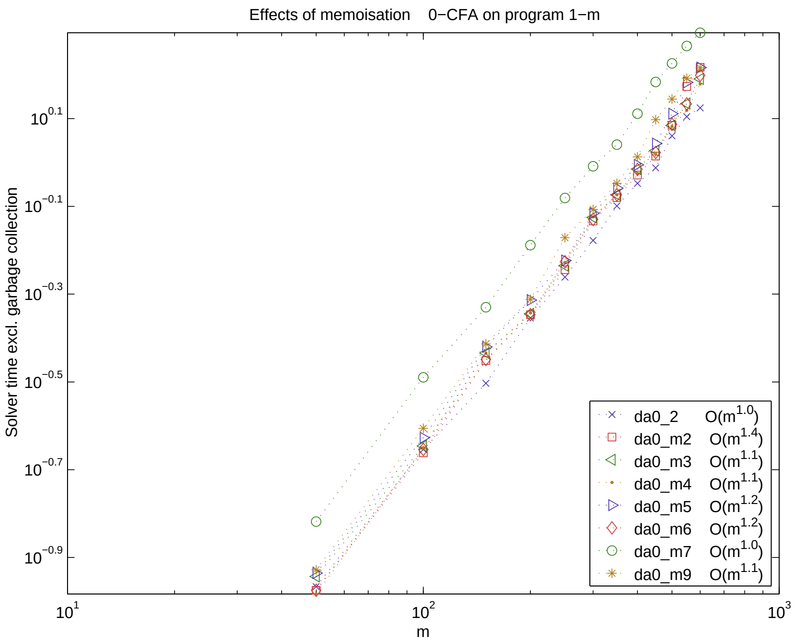
XXVI 0-CFA – Effects of Tiling

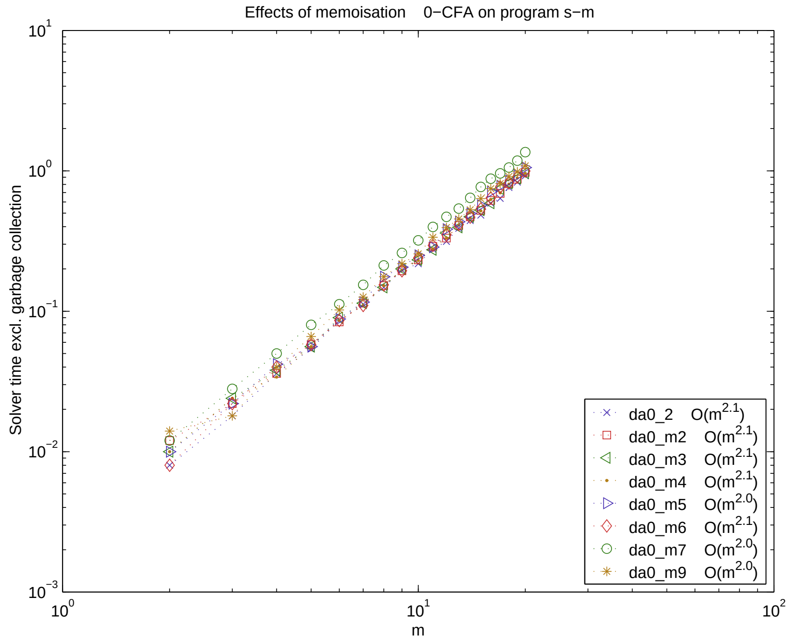


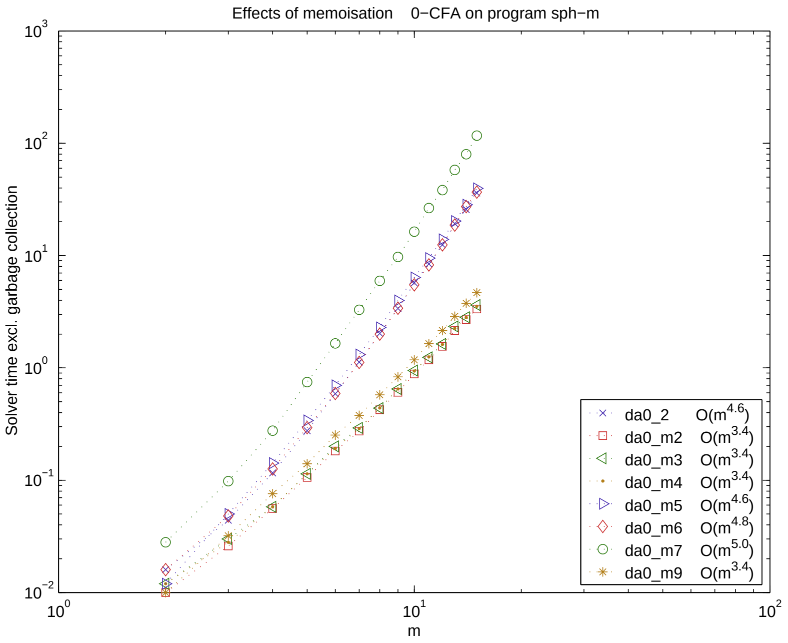



Effects of tiling   0−CFA on program sph−m
Solver time excl. garbage collection
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{0}
10^{1}
10^{2}
m
da0_2 O($m^{4.6}$)
da0_t1 O($m^{4.6}$)
da0_t2 O($m^{3.4}$)
da0_t3 O($m^{3.5}$)
da0_t4 O($m^{3.4}$)
da0_t5 O($m^{4.5}$)
da0_t6 O($m^{4.7}$)
da0_t7 O($m^{4.7}$)

XXVII 0-CFA – Effects of Memoisation

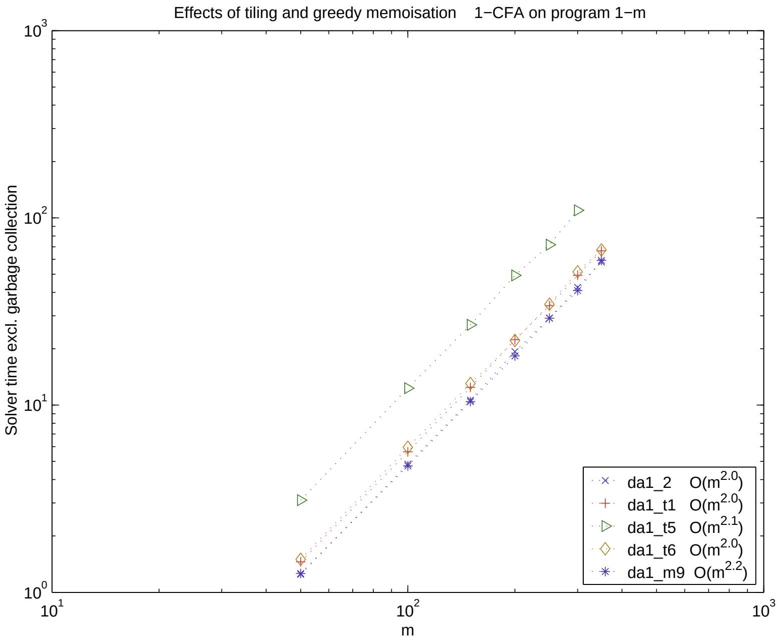


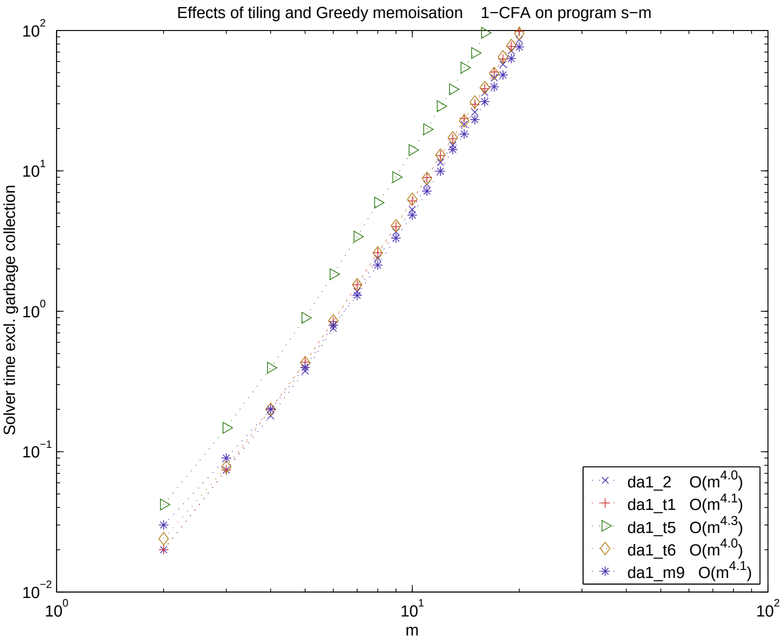


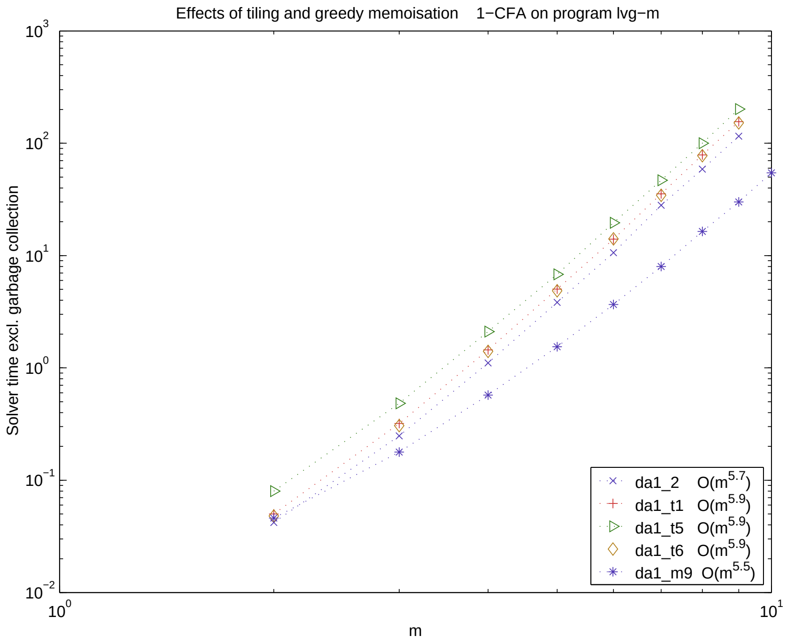


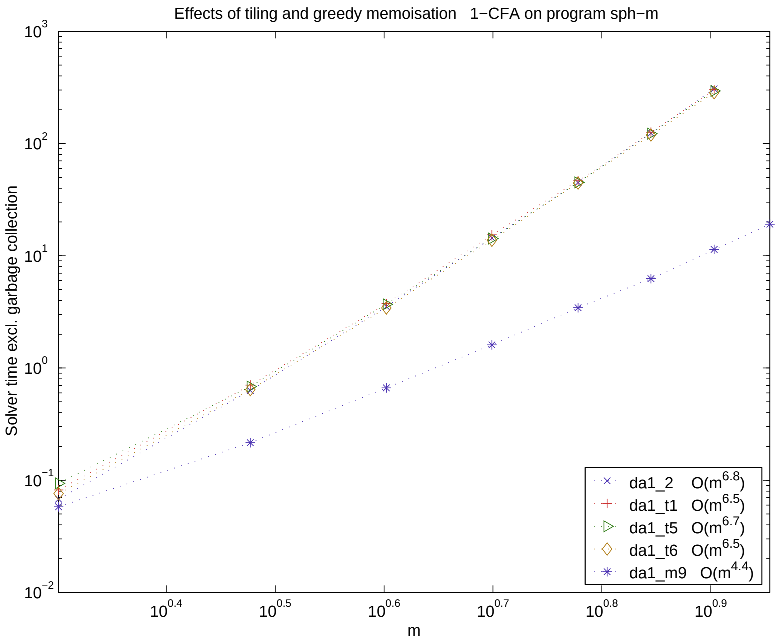
Effects of memoisation 0−CFA on program sph−m

XXVIII 1-CFA – Effects of Tiling and Greedy memoisation






Effects of tiling and greedy memoisation 1−CFA on program lvg−m
Solver time excl. garbage collection
m
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{0}
10^{1}
da1_2 $O(m^{5.7})$
da1_t1 $O(m^{5.9})$
da1_t5 $O(m^{5.9})$
da1_t6 $O(m^{5.9})$
da1_m9 $O(m^{5.5})$


Effects of tiling and greedy memoisation 1−CFA on program sph−m
Solver time excl. garbage collection
m
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
$10^{0.4}$
$10^{0.5}$
$10^{0.6}$
$10^{0.7}$
$10^{0.8}$
$10^{0.9}$
da1_2 $O(m^{6.8})$
da1_t1 $O(m^{6.5})$
da1_t5 $O(m^{6.7})$
da1_t6 $O(m^{6.5})$
da1_m9 $O(m^{4.4})$