Reachability Analysis In Graph Transformation Systems

Technical Report

by

Abhisekh Sankaran

under the guidance of

Prof. Supratik Chakraborty

Computer Science & Engineering Department
Indian Institute of Technology
Bombay

2007

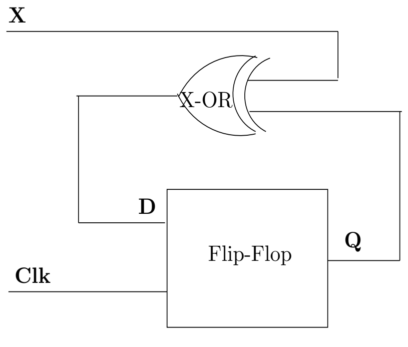
Contents

Chapter 1

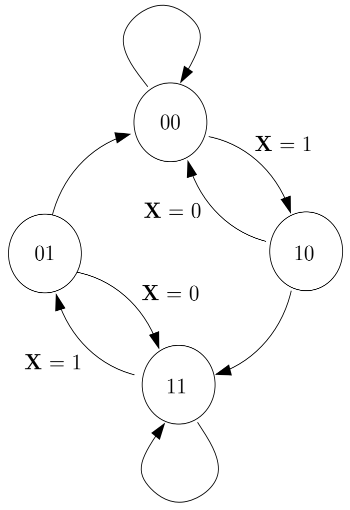
Introduction

State transition systems are a useful mathematical model as they can be used to express the behaviour of wide variety of systems. Often it is desired to check properties that must be true at any point of evolution of the system. Such properties are called *safety* properties. The checking of these properties in the system can be translated to a *reachability* analysis in the state-transition model of the system, where the reachability of states of the system violating the property is checked. Thus reachability analysis plays a central role in the verification of the behaviour of systems.

For finite state systems, a variety of techniques have been developed to deal with the problem of reachability. Usually these systems are hardware systems though there are many software systems which are finite state. A simple example of a finite state system is shown below.



Flip-Flop Circuit



State-transition graph

It consists of a clocked flip-flop whose output (**Q**) is XOR-ed with an external input **X** to form the input (**D**) of the flip-flop. The delay of the XOR gate is negligible and that of the flip-flop is about the same as the width of the clock. A state is defined by variables **D** and **Q**. The 4 states of the circuit are shown in the state-transition graph above. Input **X** causes transitions between states. An unlabeled edge means that the **Clk** input itself causes the transition. In this graph, the question of reachability though is a trivial one; every state is reachable from every other.

Reachability in finite state systems is known to be decidable. Explicit representation techniques which operate on the state-transition graph of the system, include depth-first searches(DFS) or breadth-first searches(BFS) for the property violating states, CTL model checking on explicit representation of Kripke structures and LTL model checking by Tableau construction[WOL87]. However several realistic systems have states whose number runs into billions and hence explicit representation of the state-transition graph becomes impractical. Such systems are implicitly represented by their initial states and a transition relation which tells the next states of the system given any present state. BDD-based techniques like symbolic model checking[KLM92] take this implicit representation and use characteristic functions to represent sets of states and transition relations and do symbolic DFS or BFS or symbolic fixpoint computations. There are tools for performing automated reachability analysis of systems. Two of these are SPIN[MP86] and NuSMV[CCG02]. There are tools which do reachability analysis in systems where timing is involved. Some of these include Kronos[DOTY96], Uppaal, STeP[MS94] and Hytech. Recently SAT-based techniques[BCC99] are becoming very popular due to the rapid progress that SAT solving has seen in recent times. Since the first development of the basic search based algorithm proposed by Davis, Putnam, Logemann and Loveland[DPLL62] about 40 years ago, great progress has been made in SAT-solving owing to the active research and more recently, very clever heuristics. SATO, Chaff, Grasp, Berkmin, satz, cnfs are some of the extremely fast solvers that handle millions of clauses with several hundreds of thousands of variables. There are also inductive techniques that use inductive reasoning in doing reachability analysis[MSG00].

Reachability checking in infinite state systems however is in general undecidable. Such systems come primarily from the program verification domain.
A simple example of an infinite state system is the following program.

i :=0; **while**(1) i := i + 1;

where the state variable is i. A simple property to check is whether there is a state in which $6i^2 - 7i = 5$ is true. (Clearly there isn't).
Reachability analysis in infinite state systems essentially use abstraction techniques[CGJ03] in which the infinite state space of the system is partitioned into a finite number of classes. These are then regarded as *abstract* states of an abstract state-transition graph which now being finite, allows the use of the wide array of techniques for finite state reachability analysis. The abstract state-transition graph is an overapproximation of the *concrete* state-transition graph of the system so that if a bad abstract state is not reachable, then surely a

bad concrete state is also not reachable. But a path to a bad abstract state from an abstract initial state might be actually spurious when the path is mapped back to the concrete state-transition graph, in which case *refinements* to the abstraction are done. *Predicate* abstraction[FQ02] is a very important abstraction technique in which states of the concrete state space are grouped using predicates. It combines theorem proving and model checking techniques. *Control* and *Data*[KP00] abstraction are also important abstraction techniques in verification. While reachability in infinite state systems is undecidable in general, it is decidable for certain infinite state systems. *Location* reachability in push down automata is one such example. It is thus possible to make the problem of reachability decidable, by exploiting the specific properties and structure of the transition relation of special classes of infinite systems.

1.1 Graph Transforming Systems

In this report, we consider infinite state transition systems whose states are themselves graphs(possibly infinite). The graphs we consider are *typed* graphs, i.e. the nodes and edges have types associated with them. A transition from one state to another involves a graph transformation i.e. addition and deletion of nodes and edges as well as change of types associated with the nodes and edges. The transition from one state to another is caused by a *finite* number of *operations*. Each operation takes in a set of parameters and is expressed as a *finite* set of *guard-action* pairs. Each guard checks if a certain condition is true in a state and the corresponding action describes the graph transformation for that guard-action pair. The action is taken only if the guard evaluates to true. If the guard evaluates to false, then no-action is taken on any node or edge. For any operation, given a set of parameters, the next state is obtained by looking at all those guards which become true in the current state and performing cumulatively, the graph transformations described by each of the actions for which the corresponding guard became true. In general, there is non-determinism in the actions so that an operation, for a given set of parameters, can produce multiple next states. For a state, the set of all next states is then obtained by considering all operations and for each operation, all possible parameter values for it and for each parameter value, the set of all next states for that operation and those parameter values. The Graph Transformation System is then described as the set of all operations.
The system starts from a set of initial states i.e. a set of initial graphs. We are now interested in knowing whether certain *target* graphs having certain properties are reachable from the initial states of the system. In other words, we wish to know whether there exists a sequence of transitions starting from the initial graphs which leads to the target graphs of the system.

In the next chapter, we take the specific example of a simple library system which consists of *entities* whose instances are related to each other. The instances and the relations between them can be modeled as a graph where the instances form the nodes of the graph and the relations between them form the edges. This is a typed graph in which the type of a node is the entity of the instance which the node represents. There are no types for edges. Operations are performed in the system change the relations between instances of entities, i.e. operations perform graph transformations. We then introduce some safety properties for the library system which we would want the latter to satisfy.

1.2 Earlier Work On Graph Transformation Systems

Graph Grammars have been used to generate graph transformation systems. Graph grammars originated in the late 60s, motivated by considerations about pattern recognition and compiler construction. Since then the list of areas which have interacted with the development of graph grammars has grown considerably. Besides the aforementioned areas it includes software specification and development, database design, modeling of concurrent systems, and computer animation to name a few. The area of graph grammars and graph transformations generalizes formal language theory based on strings and the theory of term rewriting based on trees. Analogous to string grammars, a graph grammar GG is defined as $GG = (G_0, P)$ where G_0 is a set of initial graphs and P is a set of productions which specify how graphs are transformed. The language of the graph grammar is, just as for string grammars, defined as $L(GG) = \{G | G_0 \rightarrow^* G\}$ where $\rightarrow^*$ means zero or more applications of the productions.

Two approaches are used in graph grammars to define the productions. In the double pushout approach[CMR97], each production is defined as $L \leftarrow K \rightarrow R$. Here L, K and R are graphs and K is a subgraph of L and R. To apply this rule to a graph G, L must be a subgraph of G. Then the part of G contained in L but outside K (i.e. $L - K$) is deleted to produce a *context* graph $(G - L + K)$. Then the parts of R outside K $(R - K)$ are added to the context graph to produce the final graph which completes the application of the production to G. The 2 steps of the application of the production are called *pushouts* and hence the name *double pushout*. In the single pushout approach, each production is defined as $L \rightarrow R$. To apply this rule to a graph G, L must be a subgraph of G. Then application of the rule involves deletion of the subgraph L and replacing it with R (i.e. the rule application produces $G - L + R$). Since there is just a single rewriting step, the approach is called the *single* pushout. This approach generalises the double-pushout approach[L93].

Tools have been developed based on the above approaches. AGG[LB93] is a general development environment for algebraic graph transformation systems which follows the interpretative approach. Its special power comes from a very flexible attribution concept. AGG graphs are allowed to be attributed by any kind of Java objects. Graph transformations can be equipped with arbitrary computations on these Java objects described by a Java expression. The AGG environment consists of a graphical user interface comprising several visual editors, an interpreter, and a set of validation tools. The interpreter allows the stepwise transformation of graphs as well as rule applications as long as possible. AGG supports several kinds of validations which comprise graph parsing, consistency checking of graphs and conflict detection in concurrent transformations by critical pair analysis of graph rules. Applications of AGG include graph and rule-based modeling of software, validation of system properties by assigning a graph transformation based semantics to some system model, graph transformation based evolution of software, and the definition of visual languages based on graph grammars.

PROGRES[S97] is an integrated environment for a very high level programming language which has a formally defined semantics based on ”PROgrammed Graph Rewriting Systems”. This language supports the following programming paradigms/purposes among others: 1) Structurally object-oriented specification of attributed graph structures with multiple inheritance hierarchies and types of types (for parametric polymorphy). 2) Declarative/relational specification of derived attributes, node sets, binary relationships (directed edges), and Boolean constraints. Therefore, PROGRES can be used as 1) a very high level programming language for implementing abstract data types with a graph-like internal structure, 2) a rule-oriented language for rapid prototyping nondeterministically specified data/rule base transformations.

Chapter 2

The Library System

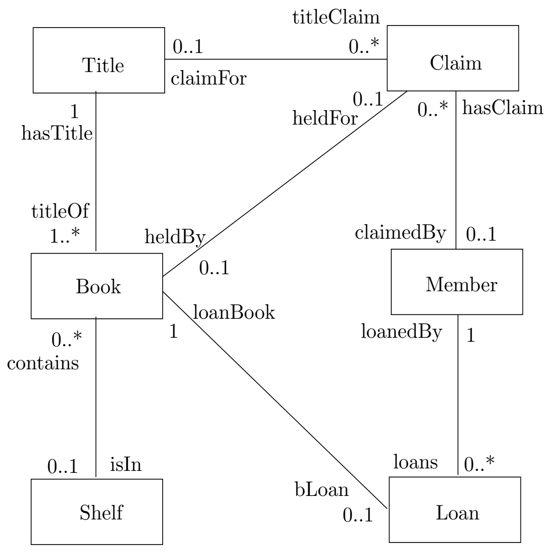


Figure 2.1: UML Class Diagram for the Library System

We first describe the library system.

The library system consists of *members* and *books*. A member can *borrow* a book kept on a *shelf*. In such a case the library *loans* the book to the member. A member can *reserve* a book by making a *claim* for *title* of the book. If a book for the claimed title exists on shelf, it is immediately reserved for the member, else the claim is kept pending until the book becomes available. A member can also borrow a book previously claimed by him. When the member *returns* a loaned book, the book goes on shelf if there are no pending claims for the title of the book. Otherwise, the book gets reserved. The library system also provides for adding more books to the library and more members to the system.

From the above description we can identify *entities* of the library system and the relations between them. These are expressed formally in a UML Class Diagram shown above.

The boxes represent the entities and the lines between them represent the *associations* between them. An association between entitites A and B specifies a relation from A to B and one from B to A. It also specifies a *multiplicity value* for each of these relations. For example, **isIn** is a relation from Book to Shelf and the multiplicity 0..1 means that when an instance of Book exists, it can be related to at most one instance of Shelf (A book needn't be on any shelf or it can be on only one shelf). But the **contains** relation having multiplicity 0..* allows an instance of Shelf to be related to any number of instances of books (A shelf can hold zero or more number of books). A multiplicity of 1 associated with the **hasTitle** relation means every instance of Book must be related to exactly one instance of a Title (A book must have exactly 1 title).

If we were to graphically depict the instances of the entities and the relations between them, we can see the library system as a graph having different kinds of nodes. Shown below is a library system having 5 books and 3 members.

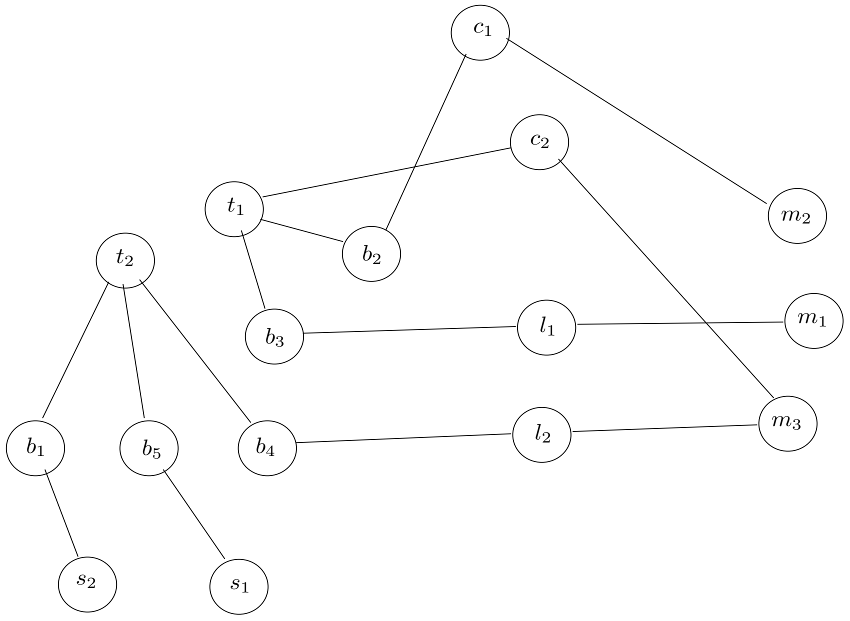


Figure 2.2: An example library system

The 2 books with titles t_1 namely b_3 and b_2 are respectively loaned to and reserved by members m_1 and m_2. Member m_3 has borrowed book b_4 having title t_2 and has a pending claim for title t_1. The other 2 books b_1 and b_5 are on shelves s_2 and s_1 respectively. c_1 and c_2 are the claim nodes while l_1 and l_2 are the loan nodes.

We can also identify *operations* taking place in the system, from our description above. We call these operations *Borrow, Reserve, Return, Addbook* and *Addmember*. With these operations taking place on the library system shown above, the graph can surely change. This motivates us to define the *state* of the library system in terms of the instances of entities and the relations between them.

We shall use First Order(FO) Logic henceforth in describing the library system and also later for formalising graph transformation systems. The problem of reachability will be formulated using FO. Note that though we have used a UML diagram to express the Library system, we are not trying to capture the semantics of the UML diagram using FO or provide any general way of capturing the semantics of UML diagrams using FO. The UML diagram just serves to show, as described in the next section, the different kinds of nodes and edges that exist in the state of the library system.

2.1 State of the library system

The entities of the library system are Book($\mathcal{B}$), Claim($\mathcal{C}$), Loan($\mathcal{L}$), Member($\mathcal{M}$), Shelf($\mathcal{S}$) and Title($\mathcal{T}$). Let $\mathcal{E} = \{\mathcal{B}, \mathcal{C}, \mathcal{L}, \mathcal{M}, \mathcal{S}, \mathcal{T}\}$. We define a state s in terms of the following predicates:-

- Node predicates:
  - $N(x)$ - This predicate is true iff the node x is present in the graph of s.
  - $X(x), X \in \mathcal{E}$ - This predicate is true iff node x represents an instance of entity X.
- Edge predicates:
  - $E(x, y)$ - This predicate is true iff an edge is present between nodes x and y.

(There are no types for the edges).
Let $\sigma_S = \{N\} \cup \{E\} \cup \{X | X \in \mathcal{E}\}$.
Then the predicates of σ_S completely define state s. Note that since there is no bound on the number of nodes in any state, the state space of the library system is infinite.

2.2 Operations of the Library System

Operations cause changes of state. Each operation takes as input some parameters and produces different graph transformations on the state under different conditions. These conditions are called *guards* and the transformations are called *actions*. An operation is expressed as a set of Guard-Action(GA) pairs. We now look at each of the operations, their semantics and their formal description in terms of GA pairs. An operation causes a transition from a state s to a state s'. We describe state s using the predicates of σ_S subscripted with s and describe state s' using the predicates of σ_S subscripted with s'.

Thus for state s, we have $\sigma_s = \{N_s\} \cup \{E_s\} \cup \{X_s | X \in \mathcal{E}\}$ and for state s', we have, $\sigma_{s'} = \{N_{s'}\} \cup \{E_{s'}\} \cup \{X_{s'} | X \in \mathcal{E}\}$.

1. *Addbook(b,t,u):*
   This operation adds a Book instance b having title as Title instance t to shelf denoted by Shelf instance u if there are no pending claims for the book . If there are pending claims, the book gets reserved.
   Viewed as a graph transforming operation, this operation adds a node for b and nodes for t and u if they are already not present in the current state. If there is no Claim node c connected to t, the $b - t$ and $t - u$ edges are added, else the $c - t$ edge is deleted and the $c - b$ and $b - t$ edges are added. We can express the GA pairs for the operation as shown below.

   Guard 1 below expresses that b is a Book node and is absent in state s (this is expressed by $\mathcal{B}_s(b) \wedge \neg N_s(b)$), t is a Title node and is present in state s ($\mathcal{T}_s(t) \wedge N_s(t)$), u is a Shelf node and is present in state s ($\mathcal{S}_s(u) \wedge N_s(u)$). $\neg\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$ expresses that there is no Claim node present in state s which has an edge to the node t. The changes produced by the action involve addition of b to s' without changing its type ($\mathcal{B}_{s'}(b) \wedge N_{s'}(b)$) and the addition of the $b - t$ and $b - u$ edges ($E_{s'}(b,t) \wedge E_{s'}(b,u)$). The other GA pairs can be understood similarly.

	Guard	*Changes produced by Action*
1.	$\mathcal{B}_s(b) \wedge \mathcal{T}_s(t) \wedge \mathcal{S}_s(u)$ $\neg N_s(b) \wedge N_s(t) \wedge N_s(u)$ $\wedge\neg\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$	$\mathcal{B}_{s'}(b) \wedge N_{s'}(b) \wedge$ $E_{s'}(b,t) \wedge E_{s'}(b,u)$
2.	$\mathcal{B}_s(b) \wedge \mathcal{T}_s(t) \wedge \mathcal{S}_s(u)$ $\neg N_s(b) \wedge \neg N_s(t) \wedge N_s(u)$ $\wedge\neg\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$	$\mathcal{B}_{s'}(b) \wedge \mathcal{T}_{s'}(t) \wedge$ $N_{s'}(b) \wedge N_{s'}(t) \wedge$ $E_{s'}(b,t) \wedge E_{s'}(b,u)$
3.	$\mathcal{B}_s(b) \wedge \mathcal{T}_s(t) \wedge \mathcal{S}_s(u)$ $\neg N_s(b) \wedge N_s(t) \wedge \neg N_s(u)$ $\wedge\neg\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$	$\mathcal{B}_{s'}(b) \wedge \mathcal{S}_{s'}(u) \wedge$ $N_{s'}(b) \wedge N_{s'}(t) \wedge$ $E_{s'}(b,t) \wedge E_{s'}(b,u)$
4.	$\mathcal{B}_s(b) \wedge \mathcal{T}_s(t) \wedge \mathcal{S}_s(u)$ $\neg N_s(b) \wedge \neg N_s(t) \wedge \neg N_s(u)$ $\wedge\neg\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$	$\mathcal{B}_{s'}(b) \wedge \mathcal{T}_{s'}(t) \wedge \mathcal{S}_{s'}(u) \wedge$ $N_{s'}(b) \wedge N_{s'}(t) \wedge N_{s'}(u) \wedge$ $E_{s'}(b,t) \wedge E_{s'}(b,u)$
5.	$\mathcal{B}_s(b) \wedge \mathcal{T}_s(t) \wedge$ $\neg N_s(b) \wedge N_s(t) \wedge$ $\wedge\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,t))$	$\mathcal{B}_{s'}(b) \wedge N_{s'}(b) \wedge$ $E_{s'}(b,t) \wedge E_{s'}(b,c) \wedge$ $\neg E_{s'}(c,t)$

In the last GA pair, if there are many $c-t$ edges, *the operation deletes exactly one $c-t$ edge and leaves others unchanged.* Hence in case of many $c-t$ edges, there are as many next states (produced by choosing different $c-t$ edges to delete).
The action above only describes the modifications to the state s. Next state s' is described completely by considering the unaffected edges and nodes of s too.

2. *Addmember(m):*
This operation adds m provided it is not already present.

	Guard	*Changes produced by Action*
1.	$\mathcal{M}_s(m) \wedge \neg N_s(m)$	$\mathcal{M}_s(m) \wedge N_s(m)$

3. *Borrow(b,m):*
Book b is loaned to member m only if either the book is on shelf or it was previously reserved by m.

Guard 1 below expresses that b is a Book node present in state s ($\mathcal{B}_s(b) \wedge N_s(b)$) and m is a Member node present in state s ($\mathcal{M}_s(m) \wedge N_s(m)$). $\exists l(\mathcal{L}_s(l) \wedge \neg N_s(l)) \wedge \exists u(\mathcal{S}_s(u) \wedge N_s(u) \wedge E_s(b,u))$ expresses that there is a Loan node l which is absent in s and there is a Shelf node u present in s which has an edge to the b node. Then the changes produced by the action involve addition of l to s' without changing its type ($\mathcal{L}_{s'}(l) \wedge N_{s'}(l)$), the addition of the $b-l$ and $l-m$ edges ($E_{s'}(b,l) \wedge E_{s'}(l,m)$) and the deletion of the $b-u$ edge ($\neg E_{s'}(b,u)$). GA pair 2 can be understood similarly.

	Guard	*Changes produced by Action*
1.	$\mathcal{B}_s(b) \wedge \mathcal{M}_s(m) \wedge N_s(b) \wedge N_s(m) \wedge$ $\exists l(\mathcal{L}_s(l) \wedge \neg N_s(l)) \wedge$ $\exists u(\mathcal{S}_s(u) \wedge N_s(u) \wedge E_s(b,u))$	$\mathcal{L}_{s'}(l) \wedge N_{s'}(l) \wedge E_{s'}(b,l) \wedge$ $E_{s'}(l,m) \wedge \neg E_{s'}(b,u)$
2.	$\mathcal{B}_s(b) \wedge \mathcal{M}_s(m) \wedge N_s(b) \wedge N_s(m) \wedge$ $\exists l(\mathcal{L}_s(l) \wedge \neg N_s(l)) \wedge$ $\exists c(\mathcal{C}_s(c) \wedge N_s(c) \wedge E_s(c,m) \wedge E_s(c,b))$	$\mathcal{L}_{s'}(l) \wedge N_{s'}(l) \wedge E_{s'}(b,l) \wedge$ $\mathcal{C}_{s'}(c) \wedge \neg N_{s'}(c) \wedge E_{s'}(l,m) \wedge$ $E_{s'}(b,l) \wedge \neg E_{s'}(c,b) \wedge \neg E_{s'}(c,m)$

4. *Reserve(t,m):*
If a book for the title t is present on shelf, it is reserved immediately. Else the claim is kept pending.
Guard 1 below expresses that t is a Title node present in state s ($\mathcal{T}_s(t) \wedge N_s(t)$) and m is a Member node present in state s ($\mathcal{M}_s(m) \wedge N_s(m)$). $\exists c(\mathcal{C}_s(c) \wedge \neg N_s(c)) \wedge \neg\exists b \exists u(\mathcal{B}_s(b) \wedge \mathcal{S}_s(u) \wedge N_s(b) \wedge N_s(u) \wedge E_s(b,t) \wedge E_s(b,u))$ expresses that there is a Claim node c which is absent in s and there is no Book node b and Shelf node u which are present in s and such that the $b-u$ and $b-t$ edges are present. The changes produced by the action involve addition of c to s' without changing its type ($\mathcal{C}_{s'}(c) \wedge N_{s'}(c)$), the addition of the $c-m$ and $c-t$ edges ($E_{s'}(c,m) \wedge E_{s'}(c,t)$). GA pair 2 can be understood similarly.

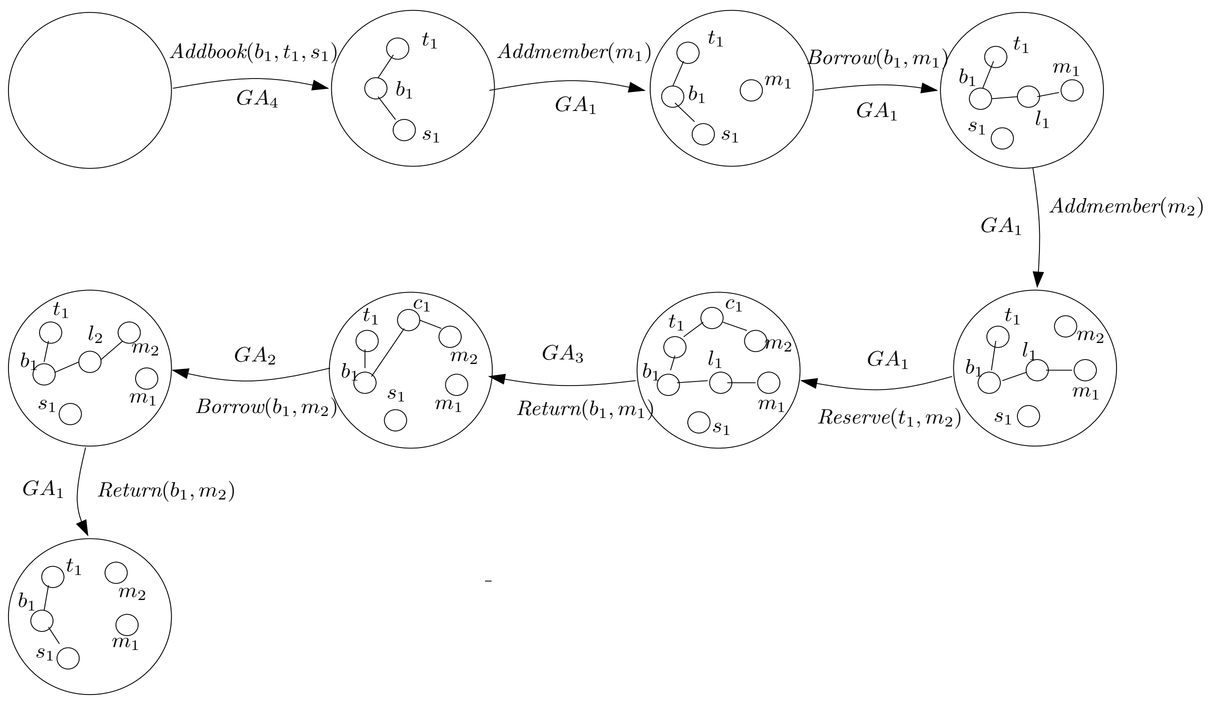
	Guard	*Changes produced by Action*
1.	$\mathcal{T}(t) \wedge N_s(t) \wedge \mathcal{M}_s(m) \wedge N_s(m) \wedge$ $\exists c(\mathcal{C}_s(c) \wedge \neg N_s(c))$ $\wedge \neg \exists b \exists u(\mathcal{B}_s(b) \wedge \mathcal{S}_s(u) \wedge N_s(b) \wedge N_s(u)$ $\wedge E_s(b,t) \wedge E_s(b,u))$	$\mathcal{C}_{s'}(c) \wedge N_{s'}(c) \wedge$ $E_{s'}(c,m) \wedge E_{s'}(c,t)$
2.	$\mathcal{T}_s(t) \wedge \mathcal{M}_s(m) \wedge N_s(t) \wedge N_s(m) \wedge$ $\exists c(\mathcal{C}_s(c) \wedge \neg N_s(c))$ $\wedge \exists b \exists u(\mathcal{B}_s(b) \wedge \mathcal{S}_s(u) \wedge N_s(b) \wedge N_s(u)$ $\wedge E_s(b,t) \wedge E_s(b,u))$	$\mathcal{C}_{s'}(c) \wedge N_{s'}(c) \wedge$ $E_{s'}(m,c) \wedge E_{s'}(c,b)$ $\neg E_{s'}(b,u)$

5. *Return(b,m):*
If book b is loaned to member m, then if there are no pending claims, the book is returned to shelf if present or a new shelf is created for the returned book. If there are pending claims for the title of the book, the book gets reserved.
Guard 1 below expresses that b is a Book node present in state s $(\mathcal{B}_s(b) \wedge N_s(b))$ and m is a Member node present in state s $(\mathcal{M}_s(m) \wedge N_s(m))$. $\exists u(\mathcal{S}_s(u) \wedge N_s(u)) \wedge \exists l(\mathcal{L}_s(l) \wedge N_s(l) \wedge E_s(b,l) \wedge E_s(l,m))$ expresses that there is a Shelf node u which is present in s and there is a Loan node l present in s which has edges to the b and m nodes. $\neg\exists t \exists c(\mathcal{T}_s(t) \wedge \mathcal{C}_s(c) \wedge N_s(t) \wedge N_s(c) \wedge E_s(b,t) \wedge E_s(c,t))$ expresses there is no Title node t and Claim node c which are present in s and such that the $b-t$ and $c-t$ edges are present. The changes produced by the action involve deletion of l to s' without changing its type $(\mathcal{L}_{s'}(l) \wedge \neg N_{s'}(l))$ and deletion of all its edges $(\neg E_{s'}(b,l) \wedge \neg E_{s'}(l,m))$ and the addition of the $b-u$ edge $(E_{s'}(b,u))$. GA pairs 2 and 3 can be understood similarly.

	Guard	*Action*
1.	$\mathcal{B}_s(b) \wedge \mathcal{M}_s(m) \wedge N_s(b) \wedge N_s(m) \wedge$ $\exists u(\mathcal{S}_s(u) \wedge N_s(u))$ $\wedge \exists l(\mathcal{L}_s(l) \wedge N_s(l) \wedge E_s(b,l)$ $\wedge E_s(l,m))$ $\wedge \neg \exists t \exists c(\mathcal{T}_s(t) \wedge \mathcal{C}_s(c) \wedge N_s(c) \wedge N_s(t)$ $\wedge E_s(b,t) \wedge E_s(c,t))$	$\mathcal{L}_{s'}(l) \wedge \neg N_{s'}(l) \wedge$ $E_{s'}(b,u) \wedge$ $\neg E_{s'}(b,l) \wedge$ $\neg E_{s'}(l,m)$
2.	$\mathcal{B}_s(b) \wedge \mathcal{M}_s(m) \wedge N_s(b) \wedge N_s(m) \wedge$ $\neg \exists u(\mathcal{S}_s(u) \wedge N_s(u))$ $\wedge \exists l(\mathcal{L}_s(l) \wedge N_s(l) \wedge E_s(b,l)$ $\wedge E_s(l,m))$ $\wedge \neg \exists t \exists c(\mathcal{T}_s(t) \wedge \mathcal{C}_s(c) \wedge N_s(c) \wedge N_s(t)$ $\wedge E_s(b,t) \wedge E_s(c,t))$	$\mathcal{L}_{s'}(l) \wedge \neg N_{s'}(l) \wedge$ $\mathcal{S}_{s'}(u) \wedge N_{s'}(u)$ $E_{s'}(b,u) \wedge$ $\neg E_{s'}(b,l) \wedge$ $\neg E_{s'}(l,m)$

$$3. \quad \begin{array}{cc} \mathcal{B}_s(b) \wedge \mathcal{M}_s(m) \wedge N_s(b) \wedge N_s(m)\wedge & \mathcal{L}_{s'}(l) \wedge \neg N_{s'}(l)\wedge \\ \wedge\exists l(\mathcal{L}_s(l) \wedge N_s(l) \wedge E_s(b,l) & E_{s'}(c,b)\wedge \\ \wedge E_s(l,m)) & \neg E_{s'}(b,l)\wedge \\ \wedge\exists t\exists c(\mathcal{T}_s(t) \wedge \mathcal{C}_s(c) \wedge N_s(c) \wedge N_s(t) & \neg E_{s'}(l,m) \\ \wedge E_s(b,t) \wedge E_s(c,t)) & \neg E_{s'}(c,t) \end{array}$$

The figure below shows how the transitions of the library system from one state to another are caused by operations taking place in the library system. Each edge is labelled with the operation and the specific GA pair of the operation involved in the transition. The library system starts from the empty graph.



2.3 Initial State Conditions and Safety Properties

The library system starts from the empty state where there are no instances of any entity. Using the predicates of σ_S, we can write the initial condition as

$$\forall x \neg N_s(x) \wedge \forall x,y \neg E_s(x,y)$$

The system evolves through the operations. While we note that a state of the system is indeed a graph, we also note that the nodes in the graph cannot be arbitrarily connected. For example, the UML diagram disallows any edge between a Shelf instance and a Loan instance or between a Member instance and a Title instance. Similarly, no Book instance must be connected to both a Loan instance and a Shelf instance otherwise it would mean that the book is simultaneously on shelf and loaned to a member. There are thus several *patterns* of interconnections between entities that the operations must not produce. We shall consider the specific problem of checking for the $\mathcal{C}$-$\mathcal{T}$-$\mathcal{B}$-$\mathcal{S}$ pattern. This pattern is present in a state if there is a Claim instance connected to a Title instance which is connected to a Book instance which in turn is connected to a Shelf instance. If this pattern is present it means that a claim for a title is kept pending even when there is a book for the title on shelf - clearly an undesirable situation. Using the predicates of σ_S, we can express this as a safety property as

$$\begin{array}{ll} \neg\exists c,t,b,s & (\mathcal{C}_s(c) \wedge \mathcal{T}_s(t) \wedge \mathcal{B}_s(b) \wedge \mathcal{S}_s(s) \wedge N_s(c) \wedge N_s(t) \wedge N_s(b) \wedge N_s(s) \wedge \\ & E_s(c,t) \wedge E_s(b,t) \wedge E_s(b,s)) \end{array}$$

We shall see later how we can check this safety property (and several other similar properties) in the library system.

While it is known that reachability analysis in infinite state systems is undecidable in general, in the next chapter, we show that reachability analysis in infinite state graph transformation systems is also, in general, undecidable.

Chapter 3

Undecidability of Reachability in Infinite State Graph Transforming Systems

Undecidability of reachability is well-known for graph-transforming systems produced by graph rewrite systems. Though we have to formally investigate how GTSes, the way we have described them, compare with GTSes produced by graph rewrite systems, there are parallels between the two and hence undecidability of reachability in our GTSes can be expected.

We show a reduction from the halting problem. The halting problem asks whether given a Turing machine M and a string w, M halts on w. The problem is known to be undecidable.
Let $M = (Q, \Sigma, \Gamma, q_1, H, \delta)$ be the given Turing machine M where Q is the set of states, $\Sigma = \{a_1, a_2, \ldots, a_r\}$ is the string alphabet, $\Gamma = \Sigma \cup \{b\}$ is the tape alphabet (b is the blank symbol), q_1 is the initial state, $H \subseteq Q$ is the set of halting states and $\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$ is the transition function (L, R denote left and right movement of the head respectively). We consider the tape of M to be semi-infinite.
We construct a graph-transformation system,S whose state space is the space of all possible configurations of M represented as graphs. Each configuration of M can be described as $y_1 y_2 \ldots y_{m-1} q_i y_m y_{m+1} \ldots y_n$ where $y_1 y_2 \ldots y_n$ denotes the string on the tape with y_n being the rightmost non-blank symbol, q_i denotes the state M is in and y_m denotes the symbol under the head.
Then define the set of types for nodes to be

$$N_t = \{A_1, A_2, \ldots, A_r, B, C, Q_1, Q_2, \ldots, Q_k\}$$

where the A_i's correspond to Σ, the Q_i's correspond to the states of Q, B corresponds to the blank symbol and C can be regarded as corresponding to the control.
Then the graph corresponding to the above configuration is shown below:

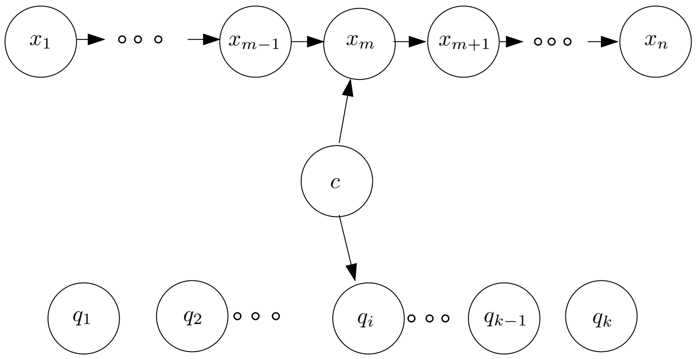


Figure 3.1: Graph corresponding to the configuration $y_1 \dots y_{m-1} q_i y_m \dots y_n$

Here $q_i (1 \leq i \leq k)$ is a node of type Q_i and c is of type C. $x_i (1 \leq i \leq n)$ is of type $A_j (1 \leq j \leq r)$ if $y_i = a_j$. x_i is of type B if $y_i = b$.
For each transition $\{q_{i'}, a_{i'}, \rho\} \in \delta(q_i, a_i)$ $(\rho \in \{L, R\})$, we construct an *operation* which has no parameters and which has just one Guard-Action(GA) pair as shown below:

- Guard
  Check if there is x_m(of type A_i) such that the edge $c \rightarrow x_m$ is present. Check if there is q_i(of type Q_i) such that the edge $c \rightarrow q_i$ is present.

- Action

  1. Delete the edge $c \rightarrow q_i$ and add the edge $c \rightarrow q_{i'}$.
  2. Delete x_m and all its incoming and outgoing edges and add an instance x'_m of $A_{i'}$.
  3. If x_m had an outgoing edge to x_{m+1}, add the edge $x'_m \rightarrow x_{m+1}$. If x_m had an incoming edge from x_{m-1}, then add the edge $x_{m-1} \rightarrow x'_m$. (This step corresponds to the overwriting of the symbol a_i with $a_{i'}$.)
  4. – $\rho = L$:
     If x_m had an incoming edge from x_{m-1}, add the edge $c \rightarrow x_{m-1}$. Else add the edge $c \rightarrow x'_m$.
     (This step corresponds to the movement of the head - if the symbol $y_m$corresponding x_m was not the leftmost symbol, the tape head is moved left so that it is now over y_{m-1}. Else the head stays where it is.)
     – $\rho = R$:
     If x_m had an outgoing edge to x_{m+1}, add the edge $c \rightarrow x_{m+1}$. Else add an instance x_{m+1} of B and add the edges $x'_m \rightarrow x_{m+1}$ and $c \rightarrow x_{m+1}$.
     (This step corresponds to the movement of the head - if the symbol $y_m$corresponding x_m was not the rightmost non-blank symbol, the tape head is moved right so that it is now over y_{m+1}. Else the head moves right and is over a blank symbol.)

The graph transformation system is then described as the set of all operations as constructed above for all the transitions in $\bigcup_{q \in Q, a \in \Sigma} \delta(q, a)$.
In the graph transformation system, the set of all next graphs for a given current graph is obtained by considering together the set of next graphs produced by each operation for each possible value of its parameters. Since each operation constructed as above has no parameters and has only one GA pair, for the (current) graph corresponding to the current configuration $y_1 \ldots y_{m-1} q a \ldots y_n$, the operation for the transition $(q', a', \rho) \in \delta(q, a)$ produces a unique next graph and this graph corresponds to the configuration obtained by taking the transition (q', a', ρ) from the current configuration. Then we can see that, for the (current) graph (corresponding to the current configuration of M), the set of all next graphs corresponds precisely to the set of all next configurations of M obtained by one move of M from the current configuration.
The initial graph is the graph corresponding to the initial configuration . If the initial configuration of M is $q_1 y_1 y_2 \ldots y_n$, then the initial graph is as shown below.

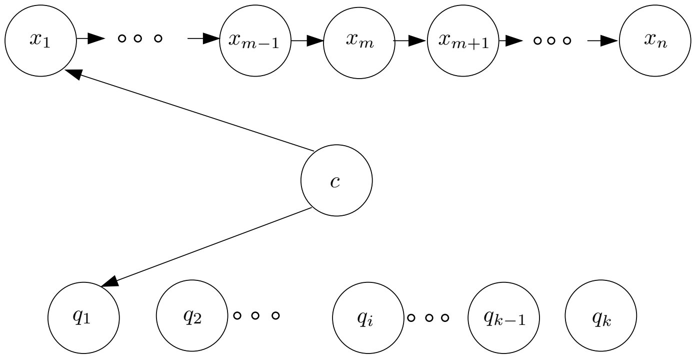


Figure 3.2: Graph corresponding to the initial configuration $q_1 y_1 \ldots y_n$

We now wish to check the reachability of the *halting* graphs in which the $c \rightarrow q_h$ edge is present where c is of type C and q_h is of type Q_h, where Q_h is any type corresponding to the halting states H. Clearly M halts on w iff any halting graph is reachable from the initial graph and the reduction is clearly a polynomial one (In fact it is a linear reduction).

$\square$

Chapter 4

Reachability Analysis Using Induction

We now look at how reachability analysis can be done using induction in a general infinite state transition system.

Consider an infinite state system whose set of states we denote by $\mathcal{S}$. Let $\mathcal{I}$ be a predicate on $\mathcal{S}$ representing all initial states. $\mathcal{T}$ is the transition relation which is a predicate on $\mathcal{S} \times \mathcal{S}$. We wish to check if a safety property $\mathcal{P}$, which is a predicate on $\mathcal{S}$, is true in all the reachable states of the system. Presented below is a procedure towards this end. This procedure draws inspiration from [MSG00] which addresses the same problem, but for a finite state system.
We first introduce some notations before explaining procedure. The notations are the same as in [MSG00]. We denote a sequence of states $s_1, s_2 \ldots s_k$ by $s_{[1\ldots k]}$. We use $\mathit{path}(s_{[1\ldots k]})$ to express that $s_{[1\ldots k]}$ forms a path in the system. Thus

$$\mathit{path}(s_{[1\ldots k]}) = \bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1})$$

While $\mathit{path}(s_{[1\ldots k]})$ expresses that $s_{[1\ldots k]}$ forms a path in the system, it does not tell us whether the path has loops or not. Loop-free paths play an important part in our procedure below, so we introduce $\mathit{loopfree}(s_{[1\ldots k]})$ to express that $s_{[1\ldots k]}$ is a loopfree path in the system. Thus

$$\mathit{loopfree}(s_{[1\ldots k]}) = \mathit{path}(s_{[1\ldots k]}) \wedge \bigwedge_{1 \le i < j \le k} s_i \neq s_j$$

We note that for any $i, 1 \le i \le k$,

$$\mathit{loopfree}(s_{[1\ldots k]}) \rightarrow \mathit{loopfree}(s_{[1\ldots i]}) \wedge \mathit{loopfree}(s_{[i+1\ldots k]})$$

Finally to express that a property $\mathcal{Z}$ is true in all the states of the sequence $s_{[1\ldots k]}$, we introduce $\mathit{all}.\mathcal{Z}(s_{[1\ldots k]})$ defined as

$$all.\mathcal{Z}(s_{[1...k]}) = \bigwedge_{i=1}^{i=k} \mathcal{Z}(s_i)$$

We now look at the procedure `InductionSolve` below. **Valid** is a procedure which takes as input a formula $F(s_1, s_2, \ldots, s_n)$ which is a boolean combination of the formulae $\mathcal{I}(s_i)(1 \leq i \leq n), \mathcal{P}(s_i)(1 \leq i \leq n), \mathcal{T}(s_i, s_j)(1 \leq i, j \leq n)$ and $s_i = s_j(1 \leq i, j \leq n)$ returns **True** if and only if the $F(s_1, s_2, \ldots, s_n)$ is true for all $(s_1, s_2, \ldots, s_n) \in \mathcal{S}^n$ i.e. if F is valid.

4.1 `InductionSolve`

```
1. If (Valid ($\mathcal{P}(s)$))
       return True
2. else if (not(Valid ($\mathcal{I}(s) \rightarrow \mathcal{P}(s)$)))
       return counterexample s.
3. else
4.     k := 1
5.     while True do
6.         If (Valid ($\big(loopfree(s_{[1...k+1]}) \wedge all.\mathcal{P}(s_{[1...k]})\big) \rightarrow \mathcal{P}(s_{k+1})$))
               return True
7.         else if (not(Valid ($\mathcal{I}(s_1)\wedge$ $loopfree(s_{[1...k+1]})\wedge$
                              $all.\neg\mathcal{I}(s_{[2...k+1]}) \wedge all.\mathcal{P}(s_{[1...k]})) \rightarrow \mathcal{P}(s_{k+1})$)))
              return counterexample $s_{[1...k+1]}$
8.         else
           k := k+1
9.     done
```

`InductionSolve`

4.2 Intuitive Idea behind `InductionSolve`

The intuitive idea of the procedure is as follows:

We first check whether $\mathcal{P}$ is true in all states of the system(line 1). If so, it is certainly true in all the reachable states and we are done. This is the base case of induction process. If this fails, then $\mathcal{P}$ is violated in some state of the system and we wish to know if it is reachable. The most basic step is to check whether the state is already an initial state(line 2). If so, we have a counterexample. Else all the initial states are $\mathcal{P}$-states (i.e. satisfy $\mathcal{P}$). We then take the inductive step of checking whether from any $\mathcal{P}$-state we can go only to another $\mathcal{P}$-state(line 6, $k = 1$) in one step. If this is true, then all the reachable states would be $\mathcal{P}$-states and we would be done. Else there is some $\mathcal{P}$-state leading to a $\neg\mathcal{P}$-state in one step and we wish to know whether it is reachable. Again we check whether this state is an initial state in the first place(line 7, $k = 1$). If so, we have a path to a state violating $\mathcal{P}$. Else all the neighbours of the initial states are also $\mathcal{P}$-states. We then take the next inductive step of checking that in a sequence of two steps is it true that if the first two states are $\mathcal{P}$-states then the third is also a $\mathcal{P}$-state(line 6 , $k = 2$). The subsequent steps are then similar to the first inductive step explained above.

4.3 Proof Of Correctness

We now formally prove the correctness of the procedure (whenever it terminates).
If the procedure returns a counterexample then it means that for some k, these are states $s_1, s_2 \ldots s_{k+1}$ such that

$$\mathcal{I}(s_1) \wedge \textit{loopfree}(s_{[1 \ldots k+1]}) \wedge \neg\mathcal{P}(s_{k+1})$$

is true. This sequence of states actually gives the path starting from the initial state to a reachable state(s_{k+1}) violating $\mathcal{P}$.
We look at the case when the procedure returns **True**. We prove that

$$\mathcal{I}(s_1) \wedge \textit{loopfree}(s_{[1 \ldots i]}) \wedge \neg\mathcal{P}(s_i)$$

is contradictory $\forall i \in \{1, 2, \ldots\}$ thus showing that if a state s satisfies $\neg\mathcal{P}(s)$, then it is not reachable from the initial states, so that all the reachable states of the system satisfy the safety property $\mathcal{P}$.

Proof:

Firstly we note that the procedure returns **True** either from line 1 or line 6. If procedure returns **True** from line 1 it means $\mathcal{P}(s)$ is true. Thus $\mathcal{I}(s) \wedge \neg\mathcal{P}(s)$ is contradictory. Let us now look at the more interesting case when procedure returns **True** from line 6. Then in that case, for some $k \geq 1$

1. $A(s_{[1 \ldots k+1]}) = \textit{loopfree}(s_{[1 \ldots k+1]}) \wedge \textit{all.}\mathcal{P}(s_{[1 \ldots k]}) \wedge \neg\mathcal{P}(s_{k+1})$
   is contradictory and

2. $\forall i \in \{1, 2, \ldots k-1\}$
   $$\begin{aligned} B(s_{[1 \ldots k+1]}) = \;& \mathcal{I}(s_1) \wedge \textit{loopfree}(s_{[1 \ldots k+1]}) \wedge \textit{all.}\neg\mathcal{I}(s_{[2 \ldots k+1]}) \wedge \\ & \textit{all.}\mathcal{P}(s_{[1 \ldots k]}) \wedge \neg\mathcal{P}(s_{k+1}) \end{aligned}$$
   is contradictory.

We make the important observation that, that the procedure returns **True** from line 6 itself means that **Valid**($\mathcal{I}(s) \rightarrow \mathcal{P}(s)$) returned **True** in line 2 i.e. all initial states are $\mathcal{P}$-states (i.e. they satisfy $\mathcal{P}$).
We consider two cases

1. $1 \leq i \leq k$
   Consider the following subcases:

   (a) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \neg \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   If this is satisfiable, it means there is a sequence of states $s_{[i\ldots i]}$ starting from initial state s_1 ending in $\neg\mathcal{P}$ state s_i such that some intermediate states are also initial states. Then consider the last initial state s_j $(1 < j < i)$. (Note that the $\neg\mathcal{P}$ state cannot be an initial state). Then the subsequence $s_{[j\ldots i]}$ is a sequence $s'_{[1\ldots i']}$ of length $i' = i - j + 1 < k$ such that $B(s'_{[1\ldots i']})$ is true. As seen above, this is not possible.

   (b) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \neg \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   If this is satisfiable, then consider the first state $s_j(1 < j < i \leq k)$ violating $\mathcal{P}$. Then $B(s_{[1\ldots j]})$ is true which is not possible.

   (c) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \neg \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \neg \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   If this is satisfiable, then consider the first state $s_j(1 < j < i \leq k)$ violating $\mathcal{P}$. If there is no intermediate initial state in the subsequence $s_{[1\ldots j]}$, then $B(s_{[1\ldots j]})$ is true which is not possible. If there is, then by case (a), we again have a contradiction.

   Putting together the subcases with (2) above, we can see that $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \neg\mathcal{P}(s_i)$ is indeed contradictory.

2. $i > k$
   We consider the following subcases.

   (a) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   If this is satisfiable, then the subsequence $s_{[i-k\ldots i]}$ is a subsequence $s'_{[1\ldots k+1]}$ such that $A(s'_{[1\ldots k+1]})$ is true, which is not possible. Thus $B(s_{[1\ldots i]})$ is false for all i.

   (b) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \neg \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   Then let the last initial state be $s_j(1 < j < i)$. Then the subsequence $s_{[j\ldots i]}$ is a sequence $s'_{[1\ldots i']}$ of some length $i' \geq 1$ for which $B(s'_{[1\ldots i']})$ is true which we have already shown to be impossible in (a). Thus the above formula is false for all i.

   (c) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \neg \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   Then consider the first state $s_j(1 < j < i)$ which violates $\mathcal{P}$. Then $B(s_{[1\ldots j]})$ is true. Contradiction. This formula too is thus false for all i.

   (d) $\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots i]}) \wedge \neg \mathit{all}.\neg\mathcal{I}(s_{[2\ldots i]}) \wedge \neg \mathit{all}.\mathcal{P}(s_{[1\ldots i-1]}) \wedge \neg\mathcal{P}(s_i)$
   Again consider the first state $s_j(1 < j < i)$ which violates $\mathcal{P}$. If there is no initial state in $s_{[2\ldots j]}$, the formula is false by subcase (a), else it is false by subcase (b). Thus the formula is false for all i.

Putting 2(a),(b),(c) and (d) together we see that the proof is complete.

$\square$

Thus in `InductionSolve`, we use a combination of inductive reasoning (line 6) and bounded model checking (line 7) to either prove that the all reachable states are $\mathcal{P}$ states or to find a path from the initial states to a property violating state.

We now note that since reachability in infinite state systems is undecidable in general, the above procedure is not guaranteed to terminate always. There are two ways of non-termination of `InductionSolve`:

1. An unbounded increase of the iterator k
2. Non-termination of the validity checker

We show now how even if the validity checker is a terminating one, we cannot avoid non-termination due the an unbounded increase of iterator k.

Consider the simple graph transformation system described in the previous section. We saw that a graph(state) of the graph transformation system corresponds to a configuration of the TM. The initial graph corresponds to the initial configuration and the property violating graphs correspond to the halting configurations of the TM. Further, for a graph s, the set of next graphs corresponds exactly to the set of next configurations of the TM for the current configuration corresponding to s. Then the validity checker does the following:

1. If the input is $\mathcal{P}(s)$ (line 1), the validity checker has to check whether at all there is a graph violating $\mathcal{P}(s)$ in the graph transformation system, i.e. whether at all there is a halting configuration. Clearly the space of all possible configurations includes the halting configuration too and hence $\neg\mathcal{P}(s)$ is satisfiable (where the satisfying assignment is the graph corresponding to the halting configuration) and hence the validity checker returns **False**.

2. If the input is $\mathcal{I}(s) \rightarrow \mathcal{P}(s)$ (line 2), the validity checker has to check whether the initial configuration is a halting configuration. This is done by simply checking if $q_1 \in H$. If yes, the validity checker returns **False** else it returns **True**.

3. If the input is
$A = \big(\mathit{loopfree}(s_{[1\ldots k+1]}) \wedge \mathit{all}.\mathcal{P}(s_{[1\ldots k]})\big) \rightarrow \mathcal{P}(s_{k+1})$ or
$B = (\mathcal{I}(s_1) \wedge \mathit{loopfree}(s_{[1\ldots k+1]}) \wedge \mathit{all}.\neg\mathcal{I}(s_{[2\ldots k+1]}) \wedge \mathit{all}.\mathcal{P}(s_{[1\ldots k]})) \rightarrow \mathcal{P}(s_{k+1})$,

   (lines 6,7) the validity checker has to determine if there exists a sequence of $k+1$ configurations $s_1, s_2, \ldots, s_{k+1}$ of the TM such that the TM moves from s_i to $s_{i+1} (1 \leq i \leq k)$ and such that only s_{k+1} is a halting configuration (and in case of B whether s_1 is the only initial state). Now in k moves, the TM can move only atmost k cells to the right or k cells to the left on the tape. The validity checker then considers a tape of length $2k+1$. Then the number of configurations s_1 is $u = |Q| \times (2k+1) \times |\Gamma|^{2k+1}$. Now for each configuration, the number of next configurations of the TM is $\mathbf{max}\{|\delta(q,a)| \ |q \in Q, a \in \Gamma\}$ and this is atmost $v = |Q| \times |\Gamma| \times 2$. Then the number of sequences of configurations $s_1, s_2, \ldots, s_{k+1}$ is atmost $u \times v^k$. The validity checker enumerates out all the possible sequences of configurations $s_1, s_2, \ldots, s_{k+1}$ and checks if A resp. B is true in these sequences. If A resp. B is false for any sequence $s_1, s_2, \ldots, s_{k+1}$, then $s_1, s_2, \ldots, s_{k+1}$ is a sequence of configurations violating A resp. B and hence the validity checker returns **False**. If however A resp. B is true for all the above sequences $s_1, s_2, \ldots, s_{k+1}$, then there cannot exists a sequence of configurations $s_1, s_2, \ldots, s_{k+1}$ violating A resp. B because if there was such a sequence, then by considering only those cells of TM which were visited by the TM in the configurations $s_1, s_2, \ldots, s_{k+1}$ (the number of these cells being atmost $k+1$), we can construct a sequence of configurations $s'_1, s'_2, \ldots, s'_{k+1}$ belonging to the set of the $u.v^k$ sequences considered above. But all of the latter sequences have already been checked by the validity checker and found to satisfy A resp. B. Thus $s_1, s_2, \ldots, s_{k+1}$ cannot exist.

Thus for the graph transformation system of the previous section, we have a terminating validity checker. But we cannot put any bounds on the iterator k and expect `InductionSolve` to terminate and give the correct answer because then we would be able to decide the halting problem. The unbounded increase of k and hence the non-termination of `InductionSolve` can thus not be avoided in general.

The method of inductive reasoning and bounded model checking used in `InductionSolve` is, as we later found, very much similar to the method used in SAL(Symbolic Analysis Laboratory) SAL[NS00] is a framework for combining different tools for abstraction, program analysis, theorem proving, and model checking toward the calculation of properties (symbolic analysis) of transition systems. A key part of the SAL framework is an intermediate language for describing transition systems. This language serves as the target for translators that extract the transition system description for popular programming languages such as Esterel, Java, and Statecharts. The intermediate language also serves as a common source for driving different analysis tools through translators from the intermediate language to the input format for the tools, and from the output of these tools back to the SAL intermediate language. The main difference between `InductionSolve` and SAL is while the former uses a validity checker to do inductive reasoning or bounded model checking, the latter uses a theorem prover. Further, the latter may use human intervention during theorem proving. However by considering specific graph-transforming systems and safety properties, we can ensure that no user intervention is required for the validity checker.

In the next chapter, we look at a formalisation of graph transformation systems. We shall later use `InductionSolve` in conjunction with this formalisation to do reachability analysis in graph transforming systems.

Chapter 5

Logical Formalisation of Graph Transformation Systems

As seen earlier, Graph Transformation Systems are state-transition systems in which each state contains a graph and the transition from one state to another involves a graph transformation. The graphs we consider are *typed* graphs ,i.e. each node and each edge has a type associated with it. The type of each node is a subset of a finite set of attributes N_t for nodes ($|N_t| = \mathcal{N}_n$) and the type of each edge is a subset of a finite set of attributes E_t for edges ($|E_t| = \mathcal{N}_e$). For ease of exposition, we henceforth regard the attributes themselves as types so that a node has a set of types associated with it and similarly an edge has a set of types associated with it.
Before proceeding to formally define a state, we shall first look at some preliminaries from mathematical logic. The notation is as used in [LL98].

5.1 Preliminaries from Mathematical Logic

A vocabulary σ is a collection of constant symbols $c_1, \ldots, c_p$, predicate symbols $P_1, \ldots, P_q$ and function symbols $f_1, \ldots, f_r$. Each predicate and function symbol has an *arity* associated with it, which is the number of arguments/parameters of the symbol.
A $\sigma-$structure

$$M = \left\langle A, \left\{c_1^M, \ldots, c_p^M\right\}, \left\{P_1^M, \ldots, P_q^M\right\}, \left\{f_1^M, \ldots, f_r^M\right\}\right\rangle$$

has a universe A and interpretations of

1. each constant symbol $c_i(1 \leq i \leq p)$ as an element $c_i^M \in A$.

2. each predicate symbol $P_i(1 \leq i \leq q)$ having arity k_i as a k_i-ary relation on A, i.e. $P_i^M \subseteq A^{k_i}$

3. each function symbol $f_i(1 \leq i \leq r)$ having arity k_i as a k_i-ary function from A^{k_i} to A, i.e. $f_i^M : A^{k_i} \rightarrow A$.

A vocabulary is called *relational* if it contains only predicate symbols and constant symbols and no function symbols and is called *purely relational* if it contains only predicate symbols and no constant or function symbols.
We now look at First Order (FO) formulae, free and bound variables and semantics of FO formulae.

Variables are denoted as $x, y, z, \ldots$ with subscripts and superscripts. Then terms and FO formulae are defined as below.

- Each variable x is a term.
- Each constant c_i is a term.
- For a k_i-ary function symbol f_i, if $x_1, \ldots, x_{k_i}$ are variables, then $f_i(x_1, \ldots, x_{k_i})$ is a term.
- If t_1, t_2 are terms, then $t_1 = t_2$ is a formula (in particular, atomic formula).
- For predicate symbol P_i of arity k_i and variables $x_1, \ldots, x_{k_i}$, $P_i(x_1, \ldots, x_{k_i})$ is a formula (in particular, atomic formula).
- If φ_1, φ_2 are formulae, then any boolean combination of these is a formula.
- If φ is a formula, $\exists x \varphi$ and $\forall x \varphi$ are formulae.

Free variables of a formula are as defined below:

- The free variable of term x is x. The constant c_i has no free variables.
- Free variables of $t_1 = t_2$ are the free variables of t_1 and t_2. Free variables of $P_i(x_1, \ldots, x_{k_i})$ and $f_i(x_1, \ldots, x_{k_i})$ are $x_1, \ldots, x_{k_i}$.
- Free variables of $\varphi_1 \wedge \varphi_2$ (and of $\varphi_1 \vee \varphi_2$) are the free variables of φ_1 and φ_2. Free variables of $\neg\varphi_1$ are the same as of φ_1.
- Free variables of $\exists x \varphi$ and $\forall x \varphi$ are the free variables of φ excluding x.

The variables which are not free are bound. A formula φ with free variables $\mathbf{x}$ is denoted as $\varphi(\mathbf{x})$.
Given a $\sigma-$ structure M, for each term t with free variables $x_1, \ldots, x_n$, given below is the inductive definition of the value of $t^M(\mathbf{a})$ where $\mathbf{a} \in A^n$. Similarly for a formula $\varphi(x_1, \ldots, x_n)$, we see below how the notion of $M \models \varphi(a_1, \ldots, a_n)$ ($\varphi(a_1, \ldots, a_n)$ is true in M) is defined.

- If t is a constant symbol c_i, then $t^M = c_i^M$.
- If t is a variable x_i, then the value of $t^M(\mathbf{a})$ is a_i.
- If $t = f_i(t_1, \ldots, t_{k_i})$, then value of $t^M(\mathbf{a})$ is $f_i^M(t_1^M(\mathbf{a}), \ldots, t_{k_i}^M(\mathbf{a}))$
- If φ is $t_1 = t_2$, then $M \models \varphi(\mathbf{a})$ iff $t_1^M(\mathbf{a}) = t_2^M(\mathbf{a})$
- If φ is $P_i(t_1, \ldots, t_{k_i})$, then $M \models \varphi(\mathbf{a})$ iff $(t_1^M(\mathbf{a}), \ldots, t_{k_i}^M(\mathbf{a})) \in P_i^M$
- If $\varphi = \neg\varphi_1$, then $M \models \varphi(\mathbf{a})$ iff $M \models \varphi_1(\mathbf{a})$ is not true.
- If $\varphi = \varphi_1 \wedge \varphi_2$, then $M \models \varphi(\mathbf{a})$ iff $M \models \varphi_1(\mathbf{a})$ and $M \models \varphi_2(\mathbf{a})$.
- If $\varphi = \varphi_1 \vee \varphi_2$, then $M \models \varphi(\mathbf{a})$ iff $M \models \varphi_1(\mathbf{a})$ or $M \models \varphi_2(\mathbf{a})$.
- If $\varphi(\mathbf{x}) = \exists y \varphi_1(y, \mathbf{x})$, then $M \models \varphi(\mathbf{a})$ iff $M \models \varphi_1(a', \mathbf{a})$ for some $a' \in A$
- If $\varphi(\mathbf{x}) = \forall y \varphi_1(y, \mathbf{x})$, then $M \models \varphi(\mathbf{a})$ iff $M \models \varphi_1(a', \mathbf{a})$ for all $a' \in A$

If σ is a relational vocabulary and $A_1 \subseteq A$, then the substructure of M generated by A_1, is a $\sigma-$structure M_1 whose universe is $A_1' = A_1 \cup \{c_i^M | 1 \leq i \leq p\}$ with $c_i^{M_1} = c_i^M$ for $1 \leq i \leq p$ and each predicate symbol P_i having arity k_i is interpreted as the restriction of P_i^M to A_1' i.e. $P_i^{M_1} = P_i^M \bigcap A_1'^{k_i}$. (Note that for a purely relational vocabulary $A_1' = A_1$.)
With the above background, we now define the state of a graph transformation system.

5.2 Defining a State

A state s is a σ_S-structure where

$$\sigma_S = \{N\} \cup \{E\} \cup \{T_N^i | 1 \leq i \leq \mathcal{N}_n\} \cup \{T_E^i | 1 \leq i \leq \mathcal{N}_e\}$$

The elements of the universe of s are nodes.

1. $N(x)$ denotes that the node x is present in s.
2. $E(x, y)$ denotes that the edge (x, y) is present in s.
3. As stated above, there is a *finite* set, N_t of types for nodes. $|N_t| = \mathcal{N}_n$. Each node has a set of types associated with it which is a subset of N_t. Then $T_N^i(x)(1 \leq i \leq \mathcal{N}_n)$ denotes that the type i is present in the set of types of node x.
4. As stated above, there is a *finite* set, E_t of types for edges. $|E_t| = \mathcal{N}_e$. Each edge has a set of types associated with it which is a subset of E_t. Then $T_E^i(x, y)(1 \leq i \leq \mathcal{N}_e)$ denotes that the type i is present in the set of types of edge (x, y).

The graph of s is then defined by the nodes and edges present in the s and by their types.

5.3 Defining A Graph Transformation System S

As earlier, we define a graph transformation as a set of parametrized operations. Each operation is a set Ω of parametrized guard-action pairs and a set of operation parameters. Formally, an operation O is a pair $O = (\Omega, \mathbf{p})$ where $\mathbf{p}$ is a vector of variables representing the operation parameters. An operation relates two states s and s' for a given set of operation parameters. A parameterized Guard-Action(GA) pair has as its parameters the operation parameters(inherited from the operation) and a set of GA parameters of its own. Intuitively, for a given set of parameters, the guard checks whether certain conditions are true of a state and if the guard holds true, actions are taken on the nodes and edges of the state. The action of each guard-action pair is described using a finite set of basic sub-actions κ. We allow non-determinism in the actions so that the GA pair can specify multiple sets of subactions on the nodes and edges of a state.
A Graph Transformation System $\mathcal{S}$ is then defined as $\mathcal{S} = (\Delta, \kappa)$ where Δ is a *finite* set of operations and κ is the *finite* set of basic sub-actions in terms of which the action of any guard-action pair is expressed.

Mathematically, given a Graph Transformation System $\mathcal{S} = (\Delta, \kappa)$, for an operation $O = (\Omega, \mathbf{p}) \in \Delta$, a guard-action pair $\xi \in \Omega$ is defined as $\xi = (\Phi_G^\xi, \Phi_A^\xi)$. We look at each of these components of ξ.

1. Φ_G^ξ :

   The guard is given by the formula $\Phi_G^\xi(\mathbf{f}_\xi, \mathbf{p})$ which is an arbitrary FO formula, with free variables $\mathbf{f}_\xi, \mathbf{p}$, in the vocabulary

   $$\sigma_s = \{N_s\} \cup \{E_s\} \cup \{T^i_{N,s} | 1 \leq i \leq \mathcal{N}_n\} \cup \{T^i_{E,s} | 1 \leq i \leq \mathcal{N}_e\}$$

   $\mathbf{f}_\xi$ denotes the GA parameters of ξ. Given values to $\mathbf{f}_\xi, \mathbf{p}$ then, the guard checks whether certain conditions are true in state s. (Note that σ_s is the same as σ_S except that each predicate of the former is subscripted with s)

2. Φ_A^ξ :

   The action is given by the formula $\Phi_A^\xi(\mathbf{f}_\xi, \mathbf{p})$. In order to define $\Phi_A^\xi(\mathbf{f}_\xi, \mathbf{p})$, we first introduce predicates for the sub-actions of κ.

   The set κ broadly consists of two kinds of sub-actions: (a) subactions on nodes κ_N and (b) subactions on edges κ_E. (Thus $\kappa = \kappa_N \cup \kappa_E$).
   The set κ_N consists of two kinds of sub-actions:

   - subactions $\kappa_{N,P} = \{\lambda_{P,1}, \lambda_{P,2}, \ldots, \lambda_{P,l}\}$ which affect the presence/absence of a node in going from state s to state s' and
   - subactions $\kappa_{N,T_i} = \{\lambda_{T_i,1}, \lambda_{T_i,2}, \ldots, \lambda_{T_i,l_i}\}$ $(1 \leq i \leq \mathcal{N}_n)$ which affect the presence/absence of type i (in the set of types associated with a node) in going from state s to state s'.

The set κ_E consists of two kinds of sub-actions:

- subactions $\kappa_{E,P} = \{\mu_{P,1}, \mu_{P,2}, \ldots, \mu_{P,m}\}$ which affect the presence/absence of an edge in going from state s to state s' and
- subactions $\kappa_{E,T_i} = \{\mu_{T_i,1}, \mu_{T_i,2}, \ldots, \mu_{T_i,m_i}\}$ $(1 \leq i \leq \mathcal{N}_e)$ which affect the presence/absence of type i (in the set of types associated with an edge) in going from state s to state s'.

The guard, for its input parameters $(\mathbf{f}_\xi, \mathbf{p})$, checks whether certain conditions are true of state s. If the guard evaluates to true, then the action *labels* each node and each edge with the sub-actions to be taken on it, for the parameters $(\mathbf{f}_\xi, \mathbf{p})$.

We introduce the following set of predicates for the sub-actions. (Below, for a vector $\mathbf{f}$, we denote the length of the vector as $|\mathbf{f}|$. Thus if $\mathbf{f} = (f_1, f_2, \ldots, f_n)$, $|\mathbf{f}| = n$.)

(a) For each sub-action $\lambda_n \in \kappa_N$ we have the predicate $X^\xi_n(\mathbf{f}, x, \mathbf{q})$ where $|\mathbf{f}| = |\mathbf{f}_\xi|$ and $|\mathbf{q}| = |\mathbf{p}|$. This predicate denotes that the action of ξ labels node x with the sub-action λ_n for the parameters $(\mathbf{f}, \mathbf{q})$.

(b) For each sub-action $\mu_e \in \kappa_E$ we have the predicate $Y^\xi_e(\mathbf{f}, x, y, \mathbf{q})$ where $|\mathbf{f}| = |\mathbf{f}_\xi|$ and $|\mathbf{q}| = |\mathbf{p}|$. This predicate denotes that the action of ξ labels edge (x, y) with the sub-action μ_e for the parameters $(\mathbf{f}, \mathbf{q})$.

For e.g.

$$X^\xi_{P,1}(\mathbf{f}_\xi, x, \mathbf{p}) \wedge \bigwedge_{i=2}^{i=l} \neg X^\xi_{P,i}(\mathbf{f}_\xi, x, \mathbf{p}) \wedge \bigwedge_{j=1}^{j=\mathcal{N}_n} X^\xi_{T_j,1}(\mathbf{f}_\xi, x, \mathbf{p}) \wedge \bigwedge_{j=1}^{j=\mathcal{N}_n} \bigwedge_{i=2}^{i=l_i} \neg X^\xi_{T_j,i}(\mathbf{f}_\xi, x, \mathbf{p})$$

means that the action of ξ for parameters $\mathbf{f}_\xi, \mathbf{p}$ labels node x with the subactions $\{\lambda_{P,1}\} \cup \{\lambda_{T_i,1} | 1 \leq i \leq \mathcal{N}_n\}$.

A set of sub-action labels specified by the action on all the nodes and edges for a set of parameter values of the GA pair forms a *labelling* for that set of parameter values. Thus a labelling produced by the action of ξ for parameters $(\mathbf{f}_\xi, \mathbf{p})$ is then the set of values of $X^\xi_n(\mathbf{f}_\xi, x, \mathbf{p})$ and $Y^\xi_e(\mathbf{f}_\xi, x, y, \mathbf{p})$ for all x, y and for all $\lambda_n, \mu_e \in \kappa$. We allow non-determinism in the action so that the action can specify multiple labellings for the same set of parameters. If the guard evaluates to false, then no sub-action is labelled on any node or edge.

Let
$\sigma_N^{all\ \xi} = \bigcup_{\xi \in \Omega} \{X^\xi_n | \lambda_n \in \kappa_N\}$; $\sigma_E^{all\ \xi} = \bigcup_{\xi \in \Omega} \{Y^\xi_e | \mu_e \in \kappa_E\}$

The action of the guard-action pair ξ is then defined as $\Phi^\xi_A(\mathbf{f}_\xi, \mathbf{p})$, an arbitrary FO formula, with free variables $\mathbf{f}_\xi, \mathbf{p}$, in the vocabulary

$$\sigma^O_A = \sigma_s \cup \sigma_N^{all\ \xi} \cup \sigma_E^{all\ \xi}$$

with the constraint that for every occurence of $X^\xi_n(\mathbf{f}, x, \mathbf{q})$ and every occurence of $Y^\xi_e(\mathbf{f}, x, y, \mathbf{q})$ in the formula, $\mathbf{q}$ is free and $\mathbf{q} = \mathbf{p}$.

Note that *the guard and the action have the same set of free variables, namely* $\boldsymbol{f}_\xi$ *and* $\boldsymbol{p}$. Note also that by this definition, we allow non-determinism in the action of a guard-action pair so that the action can specify multiple labellings for a set of GA parameters.

We now define the labelling produced by a GA pair ξ for a given set of operation parameters $\mathbf{p}$. For each $\mathbf{f}_\xi$ value, if the guard evaluates to **True**, the corresponding action is taken. Thus, given *a* labelling produced by the action for each $\mathbf{f}_\xi$ value, the labelling produced by ξ is obtained by *aggregating* these labellings i.e for each node(edge), the label produced on the node(edge) by ξ is obtained by taking the union of all the labels produced on the node(edge) by the action for all $\mathbf{f}_\xi$ values.
In case of non-determinism in the action, where for one set of GA parameter values, the action produces multiple labellings, we can see that multiple labellings are produced by the GA pair too.
We can similarly define the labelling produced by the operation for a given set of operation parameter values. Given *a* labelling produced by each GA pair of the operation, the labelling produced by the operation is obtained by aggregation i.e. for each node (edge), the label produced by the operation is obtained by taking the union of the labels produced on the node(edge) by all the GA pairs.
Again in case the GA pairs produce multiple labellings, we can see that multiple labellings are produced by the operation too.

To express the above, for each operation $O = (\Omega, \mathbf{p}) \in \Delta$, we introduce the following predicates:

For each node sub-action $\lambda_n \in \kappa_N$, we introduce the predicate $X_n^O(x, \mathbf{q})(|\mathbf{q}| = |\mathbf{p}|)$ which denotes that the operation O labels the node x with sub-action λ_n for parameters $\mathbf{q}$. Similarly for each edge sub-action $\mu_e \in \kappa_E$, we introduce the predicate $Y_e^O(x, y, \mathbf{q})(|\mathbf{q}| = |\mathbf{p}|)$ which denotes that the operation O labels the edge (x, y) with sub-action μ_e for parameters $\mathbf{q}$.
Then from the above, the operation O labels node x (edge (x, y)) with λ_n (μ_e) for parameters $\mathbf{p}$ iff the labelling produced by the action of some GA pair ξ for some GA parameter values $\mathbf{f}_\xi$ labels the node x with λ_n (edge (x, y) with μ_e). Thus we have

$$\begin{aligned}\forall x\ X_n^O(x, \mathbf{p}) \quad &\leftrightarrow \quad \bigvee_{\xi \in \Omega} \exists \mathbf{f} X_n^\xi(\mathbf{f}, x, \mathbf{p}) \\ \forall x \forall y\ Y_e^O(x, y, \mathbf{p}) \quad &\leftrightarrow \quad \bigvee_{\xi \in \Omega} \exists \mathbf{f} Y_e^\xi(\mathbf{f}, x, y, \mathbf{p})\end{aligned}$$

Denote the first formula as $Z_n(\mathbf{p})$ and the second as $Z_e(\mathbf{p})$. Let

$$Z^O(\mathbf{p}) = \bigwedge_{\lambda_n \in \kappa_N} Z_n(\mathbf{p}) \wedge \bigwedge_{\mu_e \in \kappa_E} Z_e(\mathbf{p})$$

A labelling produced by the operation for a given set of operation parameters is now translated to the next states of the operation for those set of parameters as follows.

Let $\sigma_N^O = \{X_n^O | \kappa_N \in \kappa_N\}$, $\sigma_E^O = \{Y_e^O | \kappa_e \in \kappa_E\}$
$\sigma_{s'} = \{N_{s'}\} \cup \{E_{s'}\} \cup \{T^i_{N,s'} | 1 \leq i \leq \mathcal{N}_n\} \cup \{T^i_{E,s'} | 1 \leq i \leq \mathcal{N}_e\}$
(Note that $\sigma_{s'}$ is the same as σ_S except that each predicate of the former is subscripted with s')

For each subaction $\lambda_n \in \kappa_N$, we have formula $\phi_n^O(x, \mathbf{p})$ defining the effect of the subaction for operation O i.e. $\phi_n^O(x, \mathbf{p})$ defines the condition that must be true of state s' if λ_n is taken on a node x by operation O for parameters $\mathbf{p}$. Likewise we have formulae $\phi_e^O(x, y, \mathbf{p})$ defining the effect of subaction $\mu_e \in \kappa_E$ for O. We allow $\phi_n^O(x, \mathbf{p})$ (resp. $\phi_e^O(x, y, \mathbf{p})$) to be an arbitrary FO formula, with free variables $x, \mathbf{p}$ (resp. $x, y, \mathbf{p}$), in the vocabulary

$$\sigma_{ss'}^O = \sigma_s \cup \sigma_N^O \cup \sigma_E^O \cup \sigma_{s'}$$

with the constraint that for every occurence of $X_n^O(x, \mathbf{q})$ and every occurence of $Y_e^O(x, y, \mathbf{q})$ in the formula, $\mathbf{q}$ is free and $\mathbf{q} = \mathbf{p}$.

Thus we have,

$$\begin{array}{lll} \forall x & \bigwedge_{\lambda_n \in \kappa_N} \left(X_n^O(x, \mathbf{p}) \rightarrow \phi_n^O(x, \mathbf{p})\right) & \bigwedge \\ \forall x \forall y & \bigwedge_{\mu_e \in \kappa_E} \left(Y_e^O(x, y, \mathbf{p}) \rightarrow \phi_e^O(x, y, \mathbf{p})\right) & \end{array}$$

Denote the above as $\Gamma_1^O(\mathbf{p})$.

For any node x, if no sub-action $\lambda_n \in \kappa_{N,P}$ is taken on it, its presence or absence is preserved in going from state s to state s'. Similarly, if no sub-action $\lambda_n \in \kappa_{N,T_i}$ is taken on x, the presence or absence of type i in the set of types of x is preserved in going from s to s'. Likewise for edges. Thus,

$$\begin{array}{lllll} & \forall x & \left(\bigwedge_{\lambda_n \in \kappa_N} \neg X_n^O(x, \mathbf{p})\right) & \rightarrow & (N_s(x) \leftrightarrow N_{s'}(x)) & \wedge \\ \bigwedge_{i=1}^{i=\mathcal{N}_n} & \forall x & \left(\bigwedge_{\lambda_n \in \kappa_{N,T_i}} \neg X_n^O(x, \mathbf{p})\right) & \rightarrow & \left(T^i_{N,s}(x) \leftrightarrow T^i_{N,s'}(x)\right) & \wedge \\ & \forall x \forall y & \left(\bigwedge_{\mu_e \in \kappa_E} \neg Y_e^O(x, y, \mathbf{p})\right) & \rightarrow & (E_s(x, y) \leftrightarrow E_{s'}(x, y)) & \wedge \\ \bigwedge_{j=1}^{j=\mathcal{N}_e} & \forall x \forall y & \left(\bigwedge_{\mu_e \in \kappa_{E,T_j}} \neg Y_e^O(x, y, \mathbf{p})\right) & \rightarrow & \left(T^j_{E,s}(x, y) \leftrightarrow T^j_{E,s'}(x, y)\right) & \end{array}$$

Denote the above by $\Gamma_2^O(\mathbf{p})$.
Let

$$\Gamma^O(\mathbf{p}) = \Gamma_1^O(\mathbf{p}) \wedge \Gamma_2^O(\mathbf{p})$$

Let

$$\Psi_\xi(\mathbf{f}, \mathbf{p}) \quad = \quad \left(\begin{array}{lll} \forall x & \bigwedge_{\lambda_n \in \kappa_N} \neg X_n^\xi(\mathbf{f}, x, \mathbf{p}) & \bigwedge \\ \forall x \forall y & \bigwedge_{\mu_e \in \kappa_E} \neg Y_e^\xi(\mathbf{f}, x, y, \mathbf{p}) & \end{array} \right)$$

The transition relation for the guard-action pair ξ, is then defined as

$$T^{\xi}_{ss'}(\mathbf{p}) \;=\; \left(\begin{array}{llll} \forall \mathbf{f}_\xi & \Phi^{\xi}_{G}(\mathbf{f}_\xi, \mathbf{p}) & \rightarrow & \Phi^{\xi}_{A}(\mathbf{f}_\xi, \mathbf{p}) \wedge \\ \forall \mathbf{f}_\xi & \neg\Phi^{\xi}_{G}(\mathbf{f}_\xi, \mathbf{p}) & \rightarrow & \Psi_\xi(\mathbf{f}_\xi, \mathbf{p}) \end{array} \right)$$

Thus, given a set of parameters $\mathbf{p}$, for all the values of $\mathbf{f}_\xi$ for which the condition specified by the guard $\Phi^{\xi}_{G}(\mathbf{f}_\xi, \mathbf{p})$ becomes true, the corresponding action $\Phi^{\xi}_{A}(\mathbf{f}_\xi, \mathbf{p})$ is true. For all the values of $\mathbf{f}_\xi$ for which the condition specified by $\Phi^{\xi}_{G}(\mathbf{f}_\xi, \mathbf{p})$ is false, no sub-action is taken on any node or edge for those values of $\mathbf{f}_\xi$.

We now see that given a state s, putting together $T^{\xi}_{ss'}(\mathbf{p})$ for all the GA pairs of operation O and $Z^{O}(\mathbf{p})$, we get all the labellings produced by the operation O. $\Gamma^{O}(\mathbf{p})$ translates each labelling to a set of next states of the operation for parameters $\mathbf{p}$. Then putting all of these together, we get the transition relation $T^{O}_{ss'}(\mathbf{p})$ of the operation. Thus

$$T^{O}_{ss'}(\mathbf{p}) \;=\; \left(\begin{array}{ll} \bigwedge_{\xi\in\Omega} T^{\xi}_{ss'}(\mathbf{p}) & \bigwedge \\ Z^{O}(\mathbf{p}) \wedge \Gamma^{O}(\mathbf{p}) & \end{array} \right)$$

Finally the next states produced by the Graph Transformation System is obtained by taking the union of the next states produced by all the operations for all parameter values. Thus the transition relation $T_{ss'}$ of the GTS $S = (\Delta, \kappa)$ is given as

$$T_{ss'} = \bigvee_{O=(\Omega,\mathbf{p})\in\Delta} \exists \mathbf{p} \;\; T^{O}_{ss'}(\mathbf{p})$$

Note that $T_{ss'}$ is a sentence in the vocabulary

$$\sigma_{ss'} = \bigcup_{O\in\Delta} \sigma^{O}_{ss'}$$

5.4 Transition Relation on States Defined by $\mathcal{S}$

We introduce the following notation:

For a vocabulary σ and $\sigma_1 \subseteq \sigma$, given a $\sigma-$structure M, the restriction of M to the predicates of σ_1 is denoted as $M|_{\sigma_1}$.

As noted earlier σ_s and $\sigma_{s'}$ are the same as σ_S except that the predicates of the former are subscripted with s and s' respectively. Then we can say that a σ_s-structure M defines a state s as follows:

- M and s have the same universe, A say.

- For all $x \in A$,

$$\begin{array}{rcll} N_s(x) & \leftrightarrow & N(x) & \\ T^i_{N,s}(x) & \leftrightarrow & T^i_N(x) & \qquad 1 \leq i \leq \mathcal{N}_n \end{array}$$

- For all $x, y \in A$,

$$\begin{array}{rcll} E_s(x,y) & \leftrightarrow & E(x,y) & \\ T^i_{E,s}(x,y) & \leftrightarrow & T^i_E(x,y) & \qquad 1 \leq i \leq \mathcal{N}_e \end{array}$$

(Likewise a $\sigma_{s'}$ structure also defines a state.)
Then a GTS S defines a transition relation $\mathcal{T}(s,s')$ on states as follows:

Given states s and s', $\mathcal{T}(s,s')$ is true iff there exists a $\sigma_{ss'}$−structure M such that $M|_{\sigma_s}$ defines s, $M|_{\sigma_{s'}}$ defines s' and $M \models T_{ss'}$.

In the next chapter, we look at GTSes which use some special sub-actions. In all subsequent chapters, to simplify the notation, we shall omit the superscipt O in the notation when operation O is clear from the context. Similarly we shall omit ξ in the superscripts and subscripts when GA pair ξ is clear from the context.

Chapter 6

Graph Transformation Systems With Special Sub-actions

Consider GTS $S = (\Delta, \rho)$ in which the sub-actions in ρ are as follows:

- Sub-actions on nodes ρ_N:
  - Sub-actions for presence/absence of nodes, $\rho_{N,P}$
    1. $\lambda_{P,1}$ = AddNode - adds the parameter node which is absent in the graph of state s.
    2. $\lambda_{P,2}$ = DelNode - deletes the parameter node which is present in the graph of state s.
    3. $\lambda_{P,3}$ = PresNode - preserves (the presence or absence of) the parameter node.
  - Sub-actions for type-change of nodes, $\rho_{N,T_i}, 1 \leq i \leq \mathcal{N}_n$
    1. $\lambda_{T_i,1}$ = AddNodeType$i$ - adds the type i to the set of types of the parameter node where type i is absent in the set of types of the parameter node in the state s.
    2. $\lambda_{T_i,2}$ = DelNodeType$i$ - deletes the type i from the set of types of the parameter node where type i is present in the set of types of the parameter node in the state s.
    3. $\lambda_{T_i,3}$ = PresNodeType$i$ - preserves (the presence or absence of) type i in the set of types of the parameter node.
- Sub-actions on Edges ρ_E:
  - Sub-actions for presence/absence of edges, $\rho_{E,P}$
    1. $\mu_{P,1}$ = AddEdge - adds the parameter edge which is absent in the graph of state s.

2. $\mu_{P,2}$ = DelEdge - deletes the parameter edge which is present in the graph of state s.
3. $\mu_{P,3}$ = PresEdge - preserves (the presence or absence of) the parameter edge.

– Sub-actions for type-change of edges, $\rho_{E,T_i}, 1 \leq i \leq \mathcal{N}_e$

1. $\mu_{T_i,1}$ = AddEdgeType$i$ - adds the type i to the set of types of the parameter edge where type i is absent in the set of types of the parameter edge in the state s.
2. $\mu_{T_i,2}$ = DelEdgeType$i$ - deletes the type i from the set of types of the parameter edge where type i is present in the set of types of the parameter edge in the state s.
3. $\mu_{T_i,3}$ = PresEdgeType$i$ - preserves (the presence or absence of) type i in the set of types of the parameter edge.

We can define the above sub-actions using the σ_s and the $\sigma_{s'}$ predicates. For any operation $O \in \Delta$, we can write the formulae $\phi_n^O(\lambda_n \in \rho_N)$ and $\phi_e^O(\mu_e \in \rho_E)$ as below.

We drop the n and e subscripts and the O superscript when these are clear from the context.

- Formulae for sub-actions on nodes, ρ_N:

  – Formulae for $\rho_{N,P}$:

$$\begin{aligned}
\lambda_{P,1} : \phi(x, \mathbf{p}) = \alpha_{P,1}(x, \mathbf{p}) &= (\neg N_s(x) \wedge N_{s'}(x) \wedge \mathbf{p} = \mathbf{p}) \\
\lambda_{P,2} : \phi(x, \mathbf{p}) = \alpha_{P,2}(x, \mathbf{p}) &= (N_s(x) \wedge \neg N_{s'}(x) \wedge \mathbf{p} = \mathbf{p}) \\
\lambda_{P,3} : \phi(x, \mathbf{p}) = \alpha_{P,3}(x, \mathbf{p}) &= (N_s(x) \leftrightarrow N_{s'}(x) \wedge \mathbf{p} = \mathbf{p})
\end{aligned}$$

  – Formulae for $\rho_{N,T_i}(1 \leq i \leq \mathcal{N}_n)$:

$$\begin{aligned}
\lambda_{T_i,1} : \phi(x, \mathbf{p}) = \alpha_{T_i,1}(x, \mathbf{p}) &= (\neg T^i_{N,s}(x) \wedge T^i_{N,s'}(x) \wedge \mathbf{p} = \mathbf{p}) \\
\lambda_{T_i,2} : \phi(x, \mathbf{p}) = \alpha_{T_i,2}(x, \mathbf{p}) &= (T^i_{N,s}(x) \wedge \neg T^i_{N,s'}(x) \wedge \mathbf{p} = \mathbf{p}) \\
\lambda_{T_i,3} : \phi(x, \mathbf{p}) = \alpha_{T_i,3}(x, \mathbf{p}) &= (T^i_{N,s}(x) \leftrightarrow T^i_{N,s'}(x) \wedge \mathbf{p} = \mathbf{p})
\end{aligned}$$

- Formulae for sub-actions on edges, ρ_E:

  – Formulae for $\rho_{E,P}$:

$$\begin{aligned}
\mu_{P,1} : \phi(x, y, \mathbf{p}) = \beta_{P,1}(x, \mathbf{p}) &= (\neg E_s(x, y) \wedge E_{s'}(x, y) \wedge \mathbf{p} = \mathbf{p}) \\
\mu_{P,2} : \phi(x, y, \mathbf{p}) = \beta_{P,2}(x, \mathbf{p}) &= (E_s(x, y) \wedge \neg E_{s'}(x, y) \wedge \mathbf{p} = \mathbf{p}) \\
\mu_{P,3} : \phi(x, y, \mathbf{p}) = \beta_{P,3}(x, \mathbf{p}) &= (E_s(x, y) \leftrightarrow E_{s'}(x, y) \wedge \mathbf{p} = \mathbf{p})
\end{aligned}$$

– Formulae for $\rho_{E,T_i}(1 \leq i \leq \mathcal{N}_e)$:

$$\begin{aligned}
\mu_{T_i,1} : \phi(x,y,\mathbf{p}) = \beta_{T_i,1}(x,\mathbf{p}) &= (\neg T^i_{E,s}(x,y) \wedge T^i_{E,s'}(x,y) \wedge \mathbf{p} = \mathbf{p}) \\
\mu_{T_i,2} : \phi(x,y,\mathbf{p}) = \beta_{T_i,2}(x,\mathbf{p}) &= (T^i_{E,s}(x,y) \wedge \neg T^i_{E,s'}(x,y) \wedge \mathbf{p} = \mathbf{p}) \\
\mu_{T_i,3} : \phi(x,y,\mathbf{p}) = \beta_{T_i,3}(x,\mathbf{p}) &= (T^i_{E,s}(x,y) \leftrightarrow T^i_{E,s'}(x,y) \wedge \mathbf{p} = \mathbf{p})
\end{aligned}$$

The special sub-actions defined above cover all the possibilities for a node and an edge - (a) a node can either be added, deleted or preserved (b) type $i(1 \leq i \leq \mathcal{N}_n)$ can either be added, deleted or preserved in the set of types of a node (c) an edge can either be added, deleted or preserved (d) type $i(1 \leq i \leq \mathcal{N}_e)$ can either be added, deleted or preserved in the set of types of an edge.
Consider a GTS $\mathcal{S} = (\Delta, \kappa)$. This defines a transition relation $\mathcal{T}(s,s')$ on states. The question now is whether it is possible to construct a GTS $\mathcal{S}' = (\Delta', \rho)$ such that $\mathcal{S}'$ also defines the same transition relation $\mathcal{T}(s,s')$ on states. In other words, is it the case that the transition relations on states defined by a general GTS can also be defined with a GTS that uses the special sub-actions ρ. Or is it that the former kind of GTS is more powerful than the latter kind in that the former enables us to capture transition relations on states which cannot be captured by the latter.
While this is an open question to be investigated, for the special kind of GTS described in the next chapter, the transition relation defined by it can indeed be defined by a GTS using the sub-actions ρ.
But before looking at this special kind of GTS, lets look at the GTS $S = (\Delta, \rho)$.
For any operation $O = (\Omega, \mathbf{p}) \in \Delta_\rho$, we can write Γ as

$$\Gamma(\mathbf{p}) = \Gamma_1(\mathbf{p}) \wedge \Gamma_2(\mathbf{p})$$

where $\Gamma_1(\mathbf{p})$ is given by

$$\begin{array}{rll}
\forall x & \bigwedge_{\lambda_n \in \rho_N} (\chi_n(x,\mathbf{p}) \rightarrow \alpha(x,\mathbf{p})) & \bigwedge \\
\forall x \forall y & \bigwedge_{\mu_e \in \rho_E} (\Upsilon_e(x,y,\mathbf{p}) \rightarrow \beta(x,y,\mathbf{p})) &
\end{array}$$

and $\Gamma_2(\mathbf{p})$ is given by

$$\begin{array}{rrlcll}
 & \forall x & \left(\bigwedge_{\lambda_n \in \rho_{N,P}} \neg\chi_n(x,\mathbf{p})\right) & \rightarrow & \alpha_{P,3}(x,\mathbf{p}) & \wedge \\
\bigwedge_{i=1}^{i=\mathcal{N}_n} & \forall x & \left(\bigwedge_{\lambda_n \in \rho_{N,T_i}} \neg\chi_n(x,\mathbf{p})\right) & \rightarrow & \alpha_{T_i,3}(x,\mathbf{p}) & \wedge \\
 & \forall x \forall y & \left(\bigwedge_{\mu_e \in \rho_{E,P}} \neg\Upsilon_e(x,y,\mathbf{p})\right) & \rightarrow & \beta_{P,3}(x,y,\mathbf{p}) & \wedge \\
\bigwedge_{j=1}^{j=\mathcal{N}_e} & \forall x \forall y & \left(\bigwedge_{\mu_e \in \rho_{E,T_j}} \neg\Upsilon_e(x,y,\mathbf{p})\right) & \rightarrow & \beta_{T_i,3}(x,y,\mathbf{p}) &
\end{array}$$

Now,

$$\begin{aligned}
\alpha_{P,1}(x,\mathbf{p}) &\rightarrow \neg\alpha_{P,2}(x,\mathbf{p}) \wedge \neg\alpha_{P,3}(x,\mathbf{p}) \\
&\rightarrow \neg\chi_{P,2}(x,\mathbf{p}) \wedge \neg\chi_{P,3}(x,\mathbf{p}) \wedge \neg(\textstyle\bigwedge_{i=1}^{i=3} \neg\chi_{P,i}(x,\mathbf{p})) \\
&\rightarrow \chi_{P,1}(x,\mathbf{p})
\end{aligned}$$

Thus,

$$\alpha_{P,1}(x,\mathbf{p}) \quad \leftrightarrow \quad \chi_{P,1}(x,\mathbf{p})$$

Similarly we can show

$$\begin{array}{rcl}\alpha_{P,2}(x,\mathbf{p}) & \leftrightarrow & \chi_{P,2}(x,\mathbf{p}) \\ \alpha_{P,3}(x,\mathbf{p}) & \leftrightarrow & \bigwedge_{i=1}^{i=2} \neg\chi_{P,i}(x,\mathbf{p})\end{array}$$

Then we can see that

$$N_{s'}(x) \quad \leftrightarrow \quad \chi_{P,1}(x,\mathbf{p}) \vee \Big(N_s(x) \wedge (\bigwedge_{i=1}^{i=2} \neg\chi_{P,i}(x,\mathbf{p}))\Big)$$

Now from Z, we have

$$\bigwedge_{\lambda_n \in \rho_N} \forall x\ \chi_n(x,\mathbf{p}) \leftrightarrow \bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\chi_n^{\xi}(\mathbf{f},x,\mathbf{p})$$

Then we can use the above equivalences to replace every occurence of χ_n by its corresponding equivalent expression in the RHS above. Thus we can write,

$$N_{s'}(x) \quad \leftrightarrow \quad \left(\begin{array}{l}\Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\chi_{P,1}^{\xi}(\mathbf{f},x,\mathbf{p})\Big) \ \vee \\ \Big(N_s(x) \wedge (\bigwedge_{i=1}^{i=2} \neg \Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\chi_{P,i}^{\xi}(\mathbf{f},x,\mathbf{p})\Big)\Big))\end{array}\right)$$

Call the RHS above as $\varsigma_{N,P}(x,\mathbf{p})$.
Similarly if

$$\begin{array}{rcl}\varsigma_{E,P}(x,y,\mathbf{p}) & = & \left(\begin{array}{l}\Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\Upsilon_{P,1}^{\xi}(\mathbf{f},x,y,\mathbf{p})\Big) \ \vee \\ \Big(E_s(x,y) \wedge (\bigwedge_{j=1}^{j=2} \neg \Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\Upsilon_{P,i}^{\xi}(\mathbf{f},x,y,\mathbf{p})\Big))\Big)\end{array}\right) \\ \varsigma_{N,T_i}(x,\mathbf{p}) & = & \left(\begin{array}{l}\Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\chi_{T_i,1}^{\xi}(\mathbf{f},x,\mathbf{p})\Big) \ \vee \\ \Big(T_{N,s}^{i}(x) \wedge (\bigwedge_{j=1}^{j=2} \neg \Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\chi_{T_i,j}^{\xi}(\mathbf{f},x,\mathbf{p})\Big))\Big)\end{array}\right) \quad ,1 \le i \le \mathcal{N}_n \\ \varsigma_{E,T_i}(x,y,\mathbf{p}) & = & \left(\begin{array}{l}\Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\Upsilon_{T_i,1}^{\xi}(\mathbf{f},x,y,\mathbf{p})\Big) \ \vee \\ \Big(T_{E,s}^{i}(x,y) \wedge (\bigwedge_{j=1}^{j=2} \neg \Big(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\,\Upsilon_{T_i,j}^{\xi}(\mathbf{f},x,y,\mathbf{p})\Big))\Big)\end{array}\right) \quad ,1 \le i \le \mathcal{N}_e\end{array}$$

then, we can see that for any operation $O \in \Delta$,

$$\begin{array}{rcll}E_{s'}(x,y) & \leftrightarrow & \varsigma_{E,P}(x,y,\mathbf{p}) & \\ T_{N,s'}^{i}(x,y) & \leftrightarrow & \varsigma_{N,T_i}(x,\mathbf{p}) & 1 \le i \le \mathcal{N}_n \\ T_{E,s'}^{i}(x,y) & \leftrightarrow & \varsigma_{E,T_i}(x,y,\mathbf{p}) & 1 \le i \le \mathcal{N}_e\end{array}$$

The above expressions will help us in the next chapter. We will also need the following result in the next chapter.

Claim 1:

Let σ be a relational vocabulary and $\sigma_1 \subseteq \sigma$. Let $\varphi(\mathbf{f})$ be a formula over σ_1. Let M be a σ-structure with universe A. Then for any $\mathbf{f}_1 \in A^{|\mathbf{f}|}$,

$$M \models \varphi(\mathbf{f}_1) \leftrightarrow M|_{\sigma_1} \models \varphi(\mathbf{f}_1)$$

Proof:

Since $\varphi(\mathbf{f})$ is a formula over σ_1, the truth of $\varphi(\mathbf{f}_1)$ is dependent only on the interpretations of the predicates of σ_1 and is independent of the interpretations of the predicates of $\sigma - \sigma_1$.

$\square$

Chapter 7

A Special Kind of Graph Transformation Systems

Firstly we recall that for any GTS, for any operation $O = (\Omega, \mathbf{p})$,

$$Z^O(\mathbf{p}) \;=\; \left(\begin{array}{llcl} \bigwedge_{\lambda_n \in \kappa_N} \forall x & X_n^O(x, \mathbf{p}) & \leftrightarrow & \bigvee_{\xi \in \Omega} \exists \mathbf{f}\, X_n^{\xi}(\mathbf{f}, x, \mathbf{p}) \;\wedge \\ \bigwedge_{\mu_e \in \kappa_E} \forall x \forall y & Y_e^O(x, y, \mathbf{p}) & \leftrightarrow & \bigvee_{\xi \in \Omega} \exists \mathbf{f}\, Y_e^{\xi}(\mathbf{f}, x, y, \mathbf{p}) \end{array} \right)$$

Using the equivalences, we can eliminate the X_n^O and the Y_e^O predicates by replacing every occurence of X_n^O and Y_e^O in Z^O and Γ^O by their corresponding equivalent expressions to get

$$\begin{aligned} C^O(\mathbf{p}) = \quad & \Gamma^O \left[\left(\bigvee_{\xi \in \Omega} \exists \mathbf{f} X_n^{\xi}(\mathbf{f}, x, \mathbf{p}) \right) | X_n^O(x, \mathbf{p}) \right] \\ & \left[\left(\bigvee_{\xi \in \Omega} \exists \mathbf{f} Y_e^{\xi}(\mathbf{f}, x, y, \mathbf{p}) \right) | Y_n^O(x, y, \mathbf{p}) \right] \end{aligned}$$

In the above $[a|b]$ denotes the replacement of every occurence of b with a. Then we can write $T_{ss'}^O(\mathbf{p})$ as

$$T_{ss'}^O(\mathbf{p}) \;=\; \bigwedge_{\xi \in \Omega} T_{ss'}^{\xi}(\mathbf{p}) \;\wedge\; C^O(\mathbf{p})$$

Consider now a GTS $S = (\Delta, \kappa)$ in which for each operation $O = (\Omega, \mathbf{p}) \in \Delta$, we have the property as described below:

For each $\xi \in \Omega$, for each $\lambda_n \in \kappa_N$ and each $\mu_e \in \kappa_E$, there exist formulae $\varphi_n^{\xi}(\mathbf{f}, x, \mathbf{p})$ and $\varphi_e^{\xi}(\mathbf{f}, x, y, \mathbf{p})$ respectively in the vocabulary

$$\sigma_s \cup \left\{ \chi_m^{\xi} | \lambda_m \in \rho_N, \xi \in \Omega \right\} \cup \left\{ \Upsilon_f^{\xi} | \mu_f \in \rho_E, \xi \in \Omega \right\}$$

φ_n^ξ and φ_e^ξ are such that every occurence of χ_m^ξ (resp. Υ_f^ξ) is of the form $\chi_m^\xi(\mathbf{f}_1, x, \mathbf{q})$ (resp. $\Upsilon_f^\xi(\mathbf{f}_1, x, y, \mathbf{q})$) where (a) $|\mathbf{f}_1| = |\mathbf{f}_\xi|$ and (b) $\mathbf{q}$ is free and $\mathbf{q} = \mathbf{p}$.
Then the following property is true.

$$\bigwedge_{\xi\in\Omega} \left(\begin{array}{lllll} \forall\mathbf{f}\forall x & \bigwedge_{\lambda_n\in\kappa_N} X_n^\xi(\mathbf{f}, x, \mathbf{p}) & \leftrightarrow & \varphi_n^\xi(\mathbf{f}, x, \mathbf{p}) \ \wedge \\ \forall\mathbf{f}\forall x\forall y & \bigwedge_{\mu_e\in\kappa_E} Y_e^\xi(\mathbf{f}, x, y, \mathbf{p}) & \leftrightarrow & \varphi_e^\xi(\mathbf{f}, x, y, \mathbf{p}) \end{array} \right)$$

Call the above as $C_1(\mathbf{p})$. Let $C_2(\mathbf{p}) =$

$$\begin{array}{llll}
\bigwedge_{\lambda_m\in\rho_N} & \forall x & \left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\chi_n^\xi(\mathbf{f}, x, \mathbf{p})\right) & \rightarrow \ \beta_n(x, \mathbf{p}) \ \wedge \\
\bigwedge_{\mu_f\in\rho_E} & \forall x\forall y & \left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\Upsilon_e^\xi(\mathbf{f}, x, \mathbf{p})\right) & \rightarrow \ \beta_e(x, y, \mathbf{p}) \ \wedge \\
 & \forall x & \left(\bigwedge_{\lambda_m\in\rho_{N,P}} \neg\left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\chi_n^\xi(\mathbf{f}, x, \mathbf{p})\right)\right) & \rightarrow \ \beta_{N,P,3}(x, \mathbf{p}) \ \wedge \\
\bigwedge_{i=1}^{i=\mathcal{N}_n} & \forall x & \left(\bigwedge_{\lambda_m\in\rho_{N,T_i}} \neg\left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\chi_n^\xi(\mathbf{f}, x, \mathbf{p})\right)\right) & \rightarrow \ \beta_{N,T_i,3}(x, \mathbf{p}) \ \wedge \\
 & \forall x\forall y & \left(\bigwedge_{\mu_f\in\rho_{E,P}} \neg\left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\Upsilon_e^\xi(\mathbf{f}, x, y, \mathbf{p})\right)\right) & \rightarrow \ \beta_{E,P,3}(x, y, \mathbf{p}) \ \wedge \\
\bigwedge_{j=1}^{j=\mathcal{N}_e} & \forall x\forall y & \left(\bigwedge_{\mu_f\in\rho_{E,T_j}} \neg\left(\bigvee_{\xi\in\Omega} \exists\mathbf{f}\Upsilon_e^\xi(\mathbf{f}, x, y, \mathbf{p})\right)\right) & \rightarrow \ \beta_{E,T_i,3}(x, y, \mathbf{p})
\end{array}$$

(Note the similarity between $C_2(\mathbf{p})$ and $C^O(\mathbf{p})$.)

Let $\mathbf{X}$ denote the vector of all the X_n^ξ predicates ($\lambda_n \in \kappa_N, \xi \in \Omega$). Similary let $\mathbf{Y}$, $\boldsymbol{\chi}$ and $\boldsymbol{\Upsilon}$ denote the vector of all Y_e^ξ, χ_m^ξ and the Υ_f^ξ predicates.
Then the condition satisfied by the GTS for each operation O is

$$\exists\mathbf{p}T_{ss'}^O(\mathbf{p}) \ \rightarrow \ \exists\mathbf{p}\,\exists\,\mathbf{X}\,\mathbf{Y}\,\boldsymbol{\chi}\,\boldsymbol{\Upsilon} \left(\begin{array}{l} T_{ss'}^O(\mathbf{p})(\mathbf{X}, \mathbf{Y}) \ \wedge \\ C_1(\mathbf{p})(\mathbf{X}, \mathbf{Y}, \boldsymbol{\chi}, \boldsymbol{\Upsilon}) \ \wedge \\ C_2(\mathbf{p})(\boldsymbol{\chi}, \boldsymbol{\Upsilon}) \end{array} \right)$$

(The above is thus a second-order condition.)

We now construct a GTS $S' = (\Delta', \rho)$ such that for each operation $O = (\Omega, \mathbf{p}) \in \Delta$, we have $O' = (\Omega', \mathbf{p}) \in \Delta'$ as follows:

- Let $\Omega = \{\xi_1, \xi_2, \ldots, \xi_r\}$. Then $\Omega' = \{\xi_1', \xi_2', \ldots, \xi_r'\}$
- $\xi' = (\Phi_G, \Phi_A)$ where
  - For $\xi' \in \{\xi_2', \ldots, \xi_r'\}$
    $\Phi_G(\mathbf{f}_{\xi'}, \mathbf{p}) = \Phi_A(\mathbf{f}_{\xi'}, \mathbf{p}) = \mathbf{True}$
    $|\mathbf{f}_{\xi'}| = |\mathbf{f}_\xi|$

– For $\xi' = \xi'_1$,
$\Phi_G(\mathbf{f}_{\xi'}, \mathbf{p}) = \mathbf{True}$
$|\mathbf{f}_{\xi'}| = |\mathbf{f}_{\xi}|$
$\Phi_A(\mathbf{f}_{\xi'}, \mathbf{p})$ is defined as follows:
Let

$$\begin{array}{rcl}
\varphi_n^{\xi'}(\mathbf{f}, x, \mathbf{p}) & = & \varphi_n^{\xi}(\mathbf{f}, x, \mathbf{p})\,[\xi'|\xi\,] \\
\varphi_e^{\xi'}(\mathbf{f}, x, y, \mathbf{p}) & = & \varphi_e^{\xi}(\mathbf{f}, x, y, \mathbf{p})\,[\xi'|\xi\,] \\
C'_2(\mathbf{p}) & = & C_2(\mathbf{p})\,[\xi'|\xi\,] \\
\varsigma'_{N,P}(x, \mathbf{p}) & = & \varsigma_{N,P}(x, \mathbf{p})\,[\xi'|\xi\,] \\
\varsigma'_{E,P}(x, y, \mathbf{p}) & = & \varsigma_{E,P}(x, y, \mathbf{p})\,[\xi'|\xi\,] \\
\varsigma'_{N,T_i}(x, \mathbf{p}) & = & \varsigma_{N,T_i}(x, \mathbf{p})\,[\xi'|\xi\,] \\
\varsigma'_{E,T_i}(x, y, \mathbf{p}) & = & \varsigma_{E,T_i}(x, y, \mathbf{p})\,[\xi'|\xi\,]
\end{array}$$

Then,

$$\begin{array}{rl}
\Phi_A(\mathbf{f}_{\xi'}, \mathbf{p}) = & (\mathbf{f}_{\xi'} = \mathbf{f}_{\xi'}) \,\wedge \\
& T^O_{ss'}(\mathbf{p})\left[\varphi_n^{\xi'}|X_n^{\xi}\,, \varphi_e^{\xi'}|Y_e^{\xi}\;\right] \\
& \qquad \left[\varsigma'_{N,P}(x, \mathbf{p})|N_{s'}(x)\right] \\
& \qquad \left[\varsigma'_{E,P}(x, y, \mathbf{p})|E_{s'}(x, y)\right] \\
& \qquad \left[\varsigma'_{N,T_i}(x, \mathbf{p})|T^i_{N,s'}(x), 1 \le i \le \mathcal{N}_n\right] \\
& \qquad \left[\varsigma'_{E,T_i}(x, y, \mathbf{p})|T^i_{E,s'}(x, y), 1 \le i \le \mathcal{N}_e\right]
\end{array}$$

The transition relation, $T^{O'}_{ss'}$ is given by

$$\begin{array}{rcl}
T^{O'}_{ss'}(\mathbf{p}) & = & \bigwedge_{\xi' \in \Omega'} T^{\xi'}_{ss'}(\mathbf{p}) \;\; \wedge C^{O'}(\mathbf{p}) \\
& = & \left(\forall \mathbf{f}_{\xi'_1}\, \Phi_A^{\xi'_1}(\mathbf{f}_{\xi'_1}, \mathbf{p})\right) \wedge C^{O'}(\mathbf{p})
\end{array}$$

Note that,

$$C^{O'}(\mathbf{p}) \;\; = \;\; C'_2(\mathbf{p})$$

Now,from $C^{O'}$, as seen in the previous chapter,

$$\begin{array}{rcl}
N_{s'}(x) & \leftrightarrow & \varsigma'_{N,P}(x, \mathbf{p}) \\
E_{s'}(x, y) & \leftrightarrow & \varsigma'_{E,P}(x, y, \mathbf{p}) \\
T^i_{N,s'}(x) & \leftrightarrow & \varsigma'_{N,T_i}(x, \mathbf{p}) \;, 1 \le i \le \mathcal{N}_n \\
T^i_{E,s'}(x, y) & \leftrightarrow & \varsigma'_{E,T_i}(x, y, \mathbf{p}) \;, 1 \le i \le \mathcal{N}_e
\end{array}$$

Using these equivalences, we can replace every occurence of $\varsigma'_{N,P}, \varsigma'_{E,P}, \varsigma'_{N,T_i}, \varsigma'_{E,T_i}$ with their corresponding LHS above, in $T^{O'}_{ss'}$ to get

$$T^{O'}_{ss'}(\mathbf{p}) \quad = \quad T^{O}_{ss'}(\mathbf{p})\left[\varphi^{\xi'}_{n}|X^{\xi}_{n}, \varphi^{\xi'}_{e}|Y^{\xi}_{e}\right] \wedge C'_2(\mathbf{p})$$

We now prove that

1. If there is a $\sigma^{O}_{ss'}$-structure M such that $M \models \exists \mathbf{p} T^{O}_{ss'}(\mathbf{p})$, then there is a $\sigma^{O'}_{ss'}$-structure M' such that $M' \models \exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p})$ and $M|_{\sigma_s \cup \sigma_{s'}} = M'|_{\sigma_s \cup \sigma_{s'}}$
2. If there is a $\sigma^{O'}_{ss'}$-structure M such that $M \models \exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p})$, then there is a $\sigma^{O}_{ss'}$-structure M' such that $M' \models \exists \mathbf{p} T^{O}_{ss'}(\mathbf{p})$ and $M|_{\sigma_s \cup \sigma_{s'}} = M'|_{\sigma_s \cup \sigma_{s'}}$

Part 1:

Let the universe of M be A. By the condition satisfied by the GTS, S, there exists $\mathbf{p}_1 \in A^{|\mathbf{p}|}$ and interpretations $\mathbf{X}_1, \mathbf{Y}_1, \boldsymbol{\chi}_1, \boldsymbol{\Upsilon}_1$ such that

$$M \quad \models \quad \begin{pmatrix} T^{O}_{ss'}(\mathbf{p})(\mathbf{X}_1, \mathbf{Y}_1) \wedge \\ C_1(\mathbf{p})(\mathbf{X}_1, \mathbf{Y}_1, \boldsymbol{\chi}_1, \boldsymbol{\Upsilon}_1) \wedge \\ C_2(\mathbf{p})(\boldsymbol{\chi}_1, \boldsymbol{\Upsilon}_1) \end{pmatrix}$$

Let $\sigma_1 = \sigma^{O}_{ss'} \cup \sigma^{O'}_{ss'}$. Construct the following σ_1-structure, M_1 :

1. It has the same universe as M, i.e. A.
2. $M_1|_{\sigma_s \cup \sigma_{s'}} = M|_{\sigma_s \cup \sigma_{s'}}$
3. The interpretations of the X^{ξ}_{n}, Y^{ξ}_{e}, $\chi^{\xi'}_{m}$ and the $\Upsilon^{\xi'}_{f}$ predicates is as follows:

$$\begin{aligned} X^{\xi}_{n}(\mathbf{f}, x, \mathbf{q}) &\leftrightarrow X^{\xi}_{1n}(\mathbf{f}, x, \mathbf{q}) \\ Y^{\xi}_{e}(\mathbf{f}, x, y, \mathbf{q}) &\leftrightarrow Y^{\xi}_{1e}(\mathbf{f}, x, y, \mathbf{q}) \\ \chi^{\xi'}_{m}(\mathbf{f}, x, \mathbf{q}) &\leftrightarrow \boldsymbol{\chi}^{\xi}_{1m}(\mathbf{f}, x, \mathbf{q}) \\ \Upsilon^{\xi'}_{f}(\mathbf{f}, x, y, \mathbf{q}) &\leftrightarrow \boldsymbol{\Upsilon}^{\xi}_{1f}(\mathbf{f}, x, y, \mathbf{q}) \end{aligned}$$

Above $\mathbf{f} \in A^{|\mathbf{f}_\xi|}, \mathbf{q} \in A^{|\mathbf{p}|}, x \in A, \lambda_n \in \kappa_N, \mu_e \in \kappa_e . \lambda_m \in \rho_N, \mu_f \in \rho_E, \xi \in \Omega$. Also X^{ξ}_{1n} is the interpretation of the X^{ξ}_{n} predicate of $\mathbf{X}_1$. Similarly for $Y^{\xi}_{1e}, \chi^{\xi}_{1m}$ and Υ^{ξ}_{1f}.
Let $C'_1(\mathbf{p}) = C_1(\mathbf{p})\left[\varphi^{\xi'}_{n}|\varphi^{\xi}_{n}, \varphi^{\xi'}_{e}|\varphi^{\xi}_{e}\right]$

Then we can see that

$$M_1 \models \left(T^{O}_{ss'}(\mathbf{p}_1) \wedge C'_1(\mathbf{p}_1) \wedge C'_2(\mathbf{p}_1)\right)$$

Using the equivalences in $C_1'(\mathbf{p}_1)$, we can write

$$\begin{aligned} M_1 &\models \left(T^{O}_{ss'}(\mathbf{p}_1) \wedge C_1'(\mathbf{p}_1) \wedge C_2'(\mathbf{p}_1)\right)\left[\varphi^{\xi'}_n | X^{\xi}_n, \varphi^{\xi'}_e | Y^{\xi}_e\right] \\ &\models \left(T^{O}_{ss'}(\mathbf{p}_1)\left[\varphi^{\xi'}_n | X^{\xi}_n, \varphi^{\xi'}_e | Y^{\xi}_e\right] \wedge \mathbf{True} \wedge C_2'(\mathbf{p}_1)\right) \\ &\models T^{O'}_{ss'}(\mathbf{p}_1) \\ &\models \exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p}) \end{aligned}$$

Now the formula $\exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p})$ is over the vocabulary $\sigma^{O'}_{ss'} \subseteq \sigma_1$. Then using **Claim 1**, we have

$$M_1|_{\sigma^{O'}_{ss'}} \models \exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p})$$

Taking $M' = M_1|_{\sigma^{O'}_{ss'}}$, we find that M' is a $\sigma^{O'}_{ss'}$- structure such that $M'|_{\sigma_s \cup \sigma_{s'}} = M|_{\sigma_s \cup \sigma_{s'}}$

Part 2:

Consider a $\sigma^{O'}_{ss'}$-structure M with universe A such that

$$M \models \exists \mathbf{p} T^{O'}_{ss'}(\mathbf{p})$$

Then there exists $\mathbf{p}_1 \in A^{|\mathbf{p}|}$ such that

$$M \models T^{O'}_{ss'}(\mathbf{p}_1)$$

Extend M to a σ_1-structure M_1 by using the following definitions for the X^{ξ}_n and the Y^{ξ}_e predicates.

$$\begin{aligned} X^{\xi}_n(\mathbf{f}, x, \mathbf{q}) &\leftrightarrow \varphi^{\xi'}_n(\mathbf{f}, x, \mathbf{q}) \\ Y^{\xi}_e(\mathbf{f}, x, y, \mathbf{q}) &\leftrightarrow \varphi^{\xi'}_e(\mathbf{f}, x, y, \mathbf{q}) \end{aligned}$$

Above, $\mathbf{f} \in A^{|\mathbf{f}_\xi|}, \mathbf{q} \in A^{|\mathbf{p}|}, x \in A, \lambda_n \in \kappa_N, \mu_e \in \kappa_E, \xi \in \Omega$.
Then

$$M_1 \models T^{O'}_{ss'}(\mathbf{p}_1) \wedge C_1'(\mathbf{p}_1)$$

Using the equivalences in $C_1'(\mathbf{p}_1)$, we can write

$$\begin{aligned} M_1 &\models \left(T^{O'}_{ss'}(\mathbf{p}_1) \wedge C_1'(\mathbf{p}_1)\right)\left[X^{\xi}_n | \varphi^{\xi'}_n, Y^{\xi}_e | \varphi^{\xi'}_e\right] \\ &\models \left(\begin{array}{l} T^{O}_{ss'}(\mathbf{p}_1)\left[\varphi^{\xi'}_n | X^{\xi}_n, \varphi^{\xi'}_e | Y^{\xi}_e\right] \wedge \\ C_2'(\mathbf{p}_1) \wedge C_1'(\mathbf{p}_1) \end{array}\right)\left[\ X^{\xi}_n | \varphi^{\xi'}_n, Y^{\xi}_e | \varphi^{\xi'}_e\ \right] \\ &\models \left(T^{O}_{ss'}(\mathbf{p}_1) \wedge C_2'(\mathbf{p}_1) \wedge \mathbf{True}\right) \\ &\models T^{O}_{ss'}(\mathbf{p}_1) \\ &\models \exists \mathbf{p} T^{O}_{ss'}(\mathbf{p}) \end{aligned}$$

Now the formula $\exists \mathbf{p} T^O_{ss'}(\mathbf{p})$ is over the vocabulary $\sigma^O_{ss'} \subseteq \sigma_1$. Then using **Claim 1**, we have

$$M_1|_{\sigma^O_{ss'}} \models \exists \mathbf{p} T^O_{ss'}(\mathbf{p})$$

Taking $M' = M_1|_{\sigma^O_{ss'}}$, we find that M' is a $\sigma^O_{ss'}$- structure such that $M'|_{\sigma_s \cup \sigma_{s'}} = M|_{\sigma_s \cup \sigma_{s'}}$

Thus $S' = (\Delta', \rho)$ defines the same transition relation on states as $S = (\Delta, \kappa)$.

$\square$

Chapter 8

Bounded Model Property

In chapter 5, from the logical formalism for GTSes, the transition relation for the GTS was expressed by $T_{ss'}$ which is a first order(FO)logic formula. We consider safety properties and initial state conditions which also are FO expressible properties of states. Thus for a state s

- $\mathcal{P}(s)$ is true iff $s \models \varphi_P$ where φ_P is a σ_S-sentence expressing the safety property.
- $\mathcal{I}(s)$ is true iff $s \models \varphi_I$ where φ_I is a σ_S-sentence expressing the initial-state conditions.

We recall that the procedure `InductionSolve` uses a validity checker which is used to check for the validity of the formulae

$$\mathcal{I}(s_1) \rightarrow \mathcal{P}(s_1)$$
$$\mathcal{P}(s_1)$$

and for $k \geq 1$

$$\left(\bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{i=2}^{i=k+1} \neg\mathcal{I}(s_i) \wedge \bigwedge_{1 \leq i,j \leq k; i \neq j} (s_i \neq s_j) \wedge \mathcal{I}(s_1) \right) \rightarrow \mathcal{P}(s_{k+1})$$
$$\left(\bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{1 \leq i,j \leq k; i \neq j} (s_i \neq s_j) \right) \rightarrow \mathcal{P}(s_{k+1})$$

Since $T_{ss'}, \varphi_P$ and φ_I are FO formulae, the validity checker then checks the validity of a FO formula. However, FO is known to be undecidable in general. Hence, no validity checker can be guaranteed to terminate for arbitrary $T_{ss'}, \varphi_P$ and φ_I. The non-termination of the validity checker causes the non-termination of `InductionSolve`. We also saw that we cannot fix a bound on the iterator k in `InductionSolve` and expect the latter would terminate within the bound and give the correct output. The unbounded increase of the iterator then causes non-termination of `InductionSolve`. Thus there are two facets to the non-termination of `InductionSolve`:

1. Non-termination due to unbounded increase of the iterator.
2. Non-termination of the validity checker.

While (1) above cannot be avoided (whether or not the validity checker terminates) in general, we can avoid (2) by considering special classes of graph transformation systems, safety properties and initial state conditions for which we can construct a validity checker that would *decide* the validity of the formulae appearing in `InductionSolve` mentioned above.

We now observe the following:
Checking the validity of a formula is the same as checking the satisfiability of the negation of the formula. Hence considering the negation of the above formula, we have

$$\begin{aligned} IS1(0) &= \mathcal{I}(s_1) \wedge \neg\mathcal{P}(s_1) \\ IS2(0) &= \neg\mathcal{P}(s_1) \end{aligned}$$

and for $k \geq 1$

$$\begin{aligned} IS1(k) &= \bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{i=2}^{i=k+1} \neg\mathcal{I}(s_i) \wedge \bigwedge_{1\leq i,j\leq k; i\neq j} (s_i \neq s_j) \wedge \mathcal{I}(s_1) \wedge \neg\mathcal{P}(s_{k+1}) \\ IS2(k) &= \bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{1\leq i,j\leq k; i\neq j} (s_i \neq s_j) \wedge \neg\mathcal{P}(s_{k+1}) \end{aligned}$$

We then wish to check whether the above formulae are satisfiable.
Consider $IS1(k)$ above. If it is satisfiable, there exists a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$. Suppose that for every such sequence, we can find a sequence $s'_1, s'_2, \ldots, s'_{k+1}$ satisfying $IS1(k)$ such that each s'_i is bounded in size (i.e. has a bounded number of nodes and edges). If we know the bounds b_i on the sizes of each of the states s'_i, then there is only a finite number of sequences $s'_1, s'_2, \ldots, s'_{k+1}$ of states such that each s'_i is of size atmost b_i. We can exhaustively search these sequences and check if we can find any sequence which satisfies $IS1(k)$. If we find one, we know that $IS1(k)$ is satisfiable and we also would have found a satisfying assignment for $s_1, s_2, \ldots, s_{k+1}$ (namely $s'_1, s'_2, \ldots, s'_{k+1}$). Else we know that there cannot exist any sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$ because if there was, we should have been able to find a sequence $s'_1, s'_2, \ldots, s'_{k+1}$ (in which the size of s'_i is bounded by b_i) corresponding to it and satisfying $IS1(k)$. Thus the *bounded model*

property of $IS1(k)$ by which if it has a model, it also has a model of bounded size enables us to *decide* the satisfiability of $IS1(k)$. Similarly we can decide satisfiability of the other formulae if they also have the bounded model property.

We then investigate which graph transformation systems, initial state conditions and safety properties have the bounded model property, so that we can use this to get bounded models for the formulae $IS1(k)$ and $IS2(k)$ above.

In the forthcoming chapter, we shall look at 2 kinds of GTSes which have bounded models for $T_{ss'}$. We later also look some safety properties and initial state conditions which have bounded models. In this section, we look at two results that will help in proving existence of bounded models for the GTSes we shall consider.

We introduce the following notation:

If M_1 is a substructure of M, we denote it as $M_1 \subseteq M$.

Claim 2:

Let $\varphi(\mathbf{f})$ be a quantifier-free formula over the vocabulary σ. Let M be a $\sigma-$structure with universe A. Then for each $\mathbf{f}_1 \in A^{|\mathbf{f}|}$, for each $A_1 \subseteq A$ such that $\mathbf{f}_1 \in A_1^{|\mathbf{f}|}$, the substructure M_1 of M generated by A_1 satisfies

$$M_1 \models \varphi(\mathbf{f}_1) \leftrightarrow M \models \varphi(\mathbf{f}_1)$$

Proof:

Let $\mathbf{f} = (f_1, f_2, \ldots, f_n)$. For each $\mathbf{f}_1 = (f_{11}, f_{12}, \ldots, f_{1n}) \in A^{|\mathbf{f}|}$, if $\mathbf{f}_1 \in A_1^{|\mathbf{f}|}, A_1 \subseteq A$. then in the substructure M_1 of M generated by A_1, we have

$$\begin{array}{rcll} & & & P \in \sigma, P\ k-ary \\ M_1 \models P(f_{1i_1}, f_{1i_2}, \ldots, f_{1i_k}) & \leftrightarrow & M \models P(f_{1i_1}, f_{1i_2}, \ldots, f_{1i_k}), & 1 \leq i_j \leq n \\ & & & 1 \leq j \leq k \\ M_1 \models (f_{1i_1} = f_{1i_2}) & \leftrightarrow & M \models (f_{1i_1} = f_{1i_2}) & 1 \leq i_1, i_2 \leq n \end{array}$$

Since $\varphi(\mathbf{f}_1)$ is a quantifier-free formula, it is a boolean combination of the above atomic formulae. Hence

$$M_1 \models \varphi(\mathbf{f}_1) \leftrightarrow M \models \varphi(\mathbf{f}_1)$$

$\square$

Claim 3:

Let $\varphi(\mathbf{f})$ be a $\exists^*\forall^*$ formula (in prenex-normal form) over vocabulary σ with m existential quantifiers. Let M be a $\sigma-$structure with universe A such that $M \models \varphi(\mathbf{f}_1)$. Then there exists $A_1 \subseteq A$ such that

1. $\mathbf{f}_1 \in A_1^{|\mathbf{f}|}$
2. $|A_1| \leq ||\mathbf{f}_1|| + m$
3. $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 satisfies $M_2 \models \varphi(\mathbf{f}_1)$

(Above, $||\mathbf{f}_1||$ denotes the number of distinct elements in $\mathbf{f}_1$.)

Proof:

- Case 1: $m = 0$
  Then $\varphi(\mathbf{f}) = \forall y_1, y_2, \dots y_n \varphi_1(y_1, y_2, \dots, y_n, \mathbf{f}), n \geq 0$. Let A_1 be such that A_1 contains only the distinct elements of $\mathbf{f}_1$. Then $\mathbf{f}_1 \in A^{|\mathbf{f}|}, |A_1| \leq ||\mathbf{f}_1||$. Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$. Let M_2 be the substructure of M generated by A_2.

  Suppose $M_2 \models \neg\varphi(\mathbf{f}_1)$
  i.e. $M_2 \models \neg\forall y_1, y_2, \dots, y_n \varphi_1(y_1, y_2, \dots, y_n, \mathbf{f}_1)$
  Then there exist $y_{11}, y_{12}, \dots, y_{1n} \in A_2$ such that

  $$M_2 \models \neg\varphi_1(y_{11}, y_{12}, \dots, y_{1n}, \mathbf{f}_1)$$

  Since φ_1 is a quantifier free formula and M_2 is a substructure of M, by **Claim 2**,

  $$\begin{aligned} M &\models \neg\varphi_1(y_{11}, y_{12}, \dots, y_{1n}, \mathbf{f}_1) \\ &\models \exists y_1, y_2, \dots, y_n \neg\varphi_1(y_1, y_2, \dots, y_n, \mathbf{f}_1) \\ &\models \neg\varphi(\mathbf{f}_1) \end{aligned}$$

  which is a contradition. Hence $M_2 \models \varphi(\mathbf{f}_1)$.

- Case 2: $m > 0$
  Then $\varphi(\mathbf{f}) = \exists x_1, x_2, \dots, x_m \forall y_1, y_2, \dots y_n \varphi_1(x_1, x_2, \dots, x_m, y_1, y_2, \dots, y_n, \mathbf{f}), n \geq 0$.
  As $M \models \varphi(\mathbf{f}_1)$, there are elements $x_{11}, x_{12}, \dots, x_{1m} \in A$ such that

  $$M \models \forall y_1, y_2, \dots y_n \varphi_1(x_{11}, x_{12}, \dots, x_{1m}, y_1, y_2, \dots, y_n, \mathbf{f}_1)$$

As the above is a $\forall^*$ formula, by Case 1, there exists $A_1 \subseteq A$ such that $\mathbf{f}_1 \in A_1^{|\mathbf{f}|}, x_i \in A_1 (1 \leq i \leq m), |A_1| \leq ||\mathbf{f}_1|| + m$ and $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 is such that

$$\begin{aligned} M_2 &\models \forall y_1, y_2, \ldots y_n \varphi_1(x_{11}, x_{12}, \ldots, x_{1m}, y_1, y_2, \ldots, y_n, \mathbf{f}_1) \\ &\models \exists x_1, x_2, \ldots x_m, \forall y_1, y_2, \ldots y_n \varphi_1(x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n, \mathbf{f}_1) \\ &\models \varphi(\mathbf{f}_1) \end{aligned}$$

$\square$

Chapter 9

Graph Transformation Systems Having Bounded Models

9.1 Class 1

9.1.1 An Example - The Library System

We now express the library system as a Graph Transformation System. Firstly we note that for the library system

$$N_t = \{\mathit{Book}(\mathcal{B}), \mathit{Member}(\mathcal{M}), \mathit{Claim}(\mathcal{C}), \mathit{Loan}(\mathcal{L}), \mathit{Shelf}(\mathcal{S})\}$$

Then $T_N^1(x) = \mathcal{B}(x), T_N^2(x) = \mathcal{M}(x), T_N^3(x) = \mathcal{C}(x), T_N^4(x) = \mathcal{L}(x), T_N^5(x) = \mathcal{S}(x)$. There are no types for the edges so $E_t = \{\}$. (Since there are no types for the edges, there are no type change sub-actions in ρ.) The operations in the GTS for the library system correspond to the operations of *Addbook*, *Addmember*, *Reserve*, *Borrow*, and *Return*. We shall first look at the operation for *Reserve*.

Reserve(t,m):

1. If atleast one book with the title t is on shelf, then exactly one such book is removed from the shelf and reserved for the member m. As a graph transformation, if there is a $\mathcal{B}$ node adjacent to the $\mathcal{T}$ node t and adjacent to an $\mathcal{S}$ node and there exists a $\mathcal{C}$ node which is absent in the graph of the current state, then for exactly one $\mathcal{B}$ node b, one $\mathcal{S}$ node s and one $\mathcal{C}$ node c satisfying the properties just mentioned, the edges between b and s is deleted, c is added, and the edge between b and c and between c and m are added. Other nodes and edges are preserved.

2. If there is no book on the shelf with title t, the claim is kept pending. As a graph transformation, if there is no $\mathcal{B}$ node b and $\mathcal{S}$ node s such that b and t are adjacent

and b and s are adjacent and there exists a $\mathcal{C}$ node which is absent in the graph of the current state, then exactly one $\mathcal{C}$ node c which is absent in the graph of the current state is added and the $c-t$ and the $c-m$ edges are added. Other nodes and edges are preserved.

We can construct an operation $O_{Reserve} = (\{\xi_1, \xi_2\}, \mathbf{p})$ where $\mathbf{p} = (t, m)$ as follows: $\xi_1 = (\Phi_G^{\xi_1}, \Phi_A^{\xi_1}), \mathbf{f}_{\xi_1} = (b, s, c)$.
This corresponds to (1) above.
(Henceforth, when ξ is clear from the context, we shall omit it in the subscript and superscript in the notation.). Thus

$\xi_1 = (\Phi_G, \Phi_A), \mathbf{f} = (b, s, c)$

$$\Phi_G(\mathbf{f}, \mathbf{p}) = \begin{array}{l} \mathcal{M}_s(m) \wedge \mathcal{T}_s(t) \wedge \mathcal{B}_s(b) \wedge \mathcal{S}_s(s) \wedge \mathcal{C}_s(c) \wedge \\ N_s(m) \wedge N_s(t) \wedge N_s(b) \wedge N_s(s) \wedge \neg N_s(c) \wedge \\ E_s(b,t) \wedge E_s(b,s) \end{array}$$

$$\Phi_A(\mathbf{f}, \mathbf{p}) = \begin{array}{lcl} \exists \mathbf{f}_1 \left((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p}) \right) & \rightarrow & \Psi(\mathbf{f}, \mathbf{p}) \ \wedge \\ \neg \exists \mathbf{f}_1 \left((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p}) \right) & \rightarrow & \psi(\mathbf{f}, \mathbf{p}) \end{array}$$

$$\psi(\mathbf{f}, \mathbf{p}) = \begin{array}{l} \chi_{P,1}(\mathbf{f}, c, \mathbf{p}) \wedge \\ \Upsilon_{P,1}(\mathbf{f}, b, c, \mathbf{p}) \wedge \Upsilon_{P,1}(\mathbf{f}, c, m, \mathbf{p}) \wedge \Upsilon_{P,2}(\mathbf{f}, b, s, \mathbf{p}) \ \wedge \\ \forall x \left((x \neq c) \rightarrow \chi_{P,3}(\mathbf{f}, x, \mathbf{p}) \right) \ \wedge \\ \forall x, y \neg \left((x = b \wedge y = s) \bigvee (x = b \wedge y = c) \bigvee (x = c \wedge y = m) \right) \\ \quad \rightarrow \Upsilon_{P,3}(\mathbf{f}, x, y, \mathbf{p}) \ \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

In order to understand Φ_A above, we first note that

$$\psi(\mathbf{f}, \mathbf{p}) \rightarrow \chi_{P,1}(\mathbf{f}, c, \mathbf{p}) \rightarrow \neg \Psi(\mathbf{f}, \mathbf{p})$$

Now suppose $\Phi_A(\mathbf{f}_1, \mathbf{p})$ and $\Phi_A(\mathbf{f}_2, \mathbf{p})$ are true for $\mathbf{f}_1 \neq \mathbf{f}_2$. Further suppose $\psi(\mathbf{f}_1, \mathbf{p})$ and $\psi(\mathbf{f}_2, \mathbf{p})$ are true.
Consider $\mathbf{f}_1$. Since $\Phi_A(\mathbf{f}_1, \mathbf{p})$ is true we have $\exists \mathbf{f} \left((\mathbf{f} \neq \mathbf{f}_1) \wedge \psi(\mathbf{f}, \mathbf{p}) \right) \rightarrow \Psi(\mathbf{f}_1, \mathbf{p})$. Now as $\mathbf{f}_2 \neq \mathbf{f}_1$ and $\psi(\mathbf{f}_2, \mathbf{p})$, $\Psi(\mathbf{f}_1, \mathbf{p})$ is true. This is a contradiction as $\psi(\mathbf{f}_1, \mathbf{p}) \rightarrow \neg \Psi(\mathbf{f}_1, \mathbf{p})$. Thus $\psi(\mathbf{f}_1, \mathbf{p})$ and $\psi(\mathbf{f}_2, \mathbf{p})$ cannot be simultaneously true. This, coupled with the fact that

$$\Phi_A(\mathbf{f}, \mathbf{p}) \rightarrow \left(\neg \exists \mathbf{f}_1 \left((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p}) \right) \rightarrow \psi(\mathbf{f}, \mathbf{p}) \right)$$

ensures that if $\Phi_A(\mathbf{f}, \mathbf{p})$ is true for some $\mathbf{f}$, then there is exactly one $\mathbf{f}^*$ such that $\Phi_A(\mathbf{f}^*, \mathbf{p})$ and $\psi(\mathbf{f}^*, \mathbf{p})$ are true. Then from $T_{ss'}(\mathbf{p})$, we have

$$\neg \Phi_G(\mathbf{f}, \mathbf{p}) \rightarrow \Psi(\mathbf{f}, \mathbf{p}) \rightarrow \neg \psi(\mathbf{f}, \mathbf{p})$$
$$\Phi_G(\mathbf{f}, \mathbf{p}) \rightarrow \Phi_A(\mathbf{f}, \mathbf{p}) \rightarrow \exists ! \mathbf{f}^* \psi(\mathbf{f}^*, \mathbf{p})$$

($\exists!\mathbf{f}^*$ means there exists a unique $\mathbf{f}^*$)
Thus if there is some $\mathbf{f}_{\xi_1}$ for which $\Phi_G^{\xi_1}(\mathbf{f}_{\xi_1}, \mathbf{p})$ is true, then there is a unique $\mathbf{f}^*_{\xi_1}$ such that $\Phi_G^{\xi_1}(\mathbf{f}^*_{\xi_1}, \mathbf{p})$ and $\psi_{\xi_1}(\mathbf{f}^*_{\xi_1}, \mathbf{p})$ are true. *This $\mathbf{f}^*_{\xi_1}$ gives the particular b, s and c nodes that are chosen for the graph transformation as described above in (1) and $\psi_{\xi_1}(\mathbf{f}^*_{\xi_1}, \mathbf{p})$ gives the sub-actions to be taken on all nodes and edges.*

We can similarly write the GA pair ξ_2 for (2) above.

$$\xi_2 = (\Phi_G, \Phi_A), \mathbf{f} = (c).$$

$$\begin{array}{rcl}\Phi_G(\mathbf{f},\mathbf{p}) & = & \begin{array}{l}\mathcal{M}_s(m) \wedge \mathcal{T}_s(t) \wedge \mathcal{C}_s(c) \wedge N_s(m) \wedge N_s(t) \wedge \neg N_s(c)\\ \neg\exists c_1(\mathcal{C}_s(c_1) \wedge N_s(c_1) \wedge E_s(c_1, m) \wedge E_s(c,t))\\ \neg\exists b, s(\mathcal{B}_s(b) \wedge \mathcal{S}_s(s) \wedge N_s(b) \wedge N_s(s) \wedge E_s(b,s) \wedge E_s(b,t))\end{array}\end{array}$$

$$\begin{array}{rcl}\Phi_A(\mathbf{f},\mathbf{p}) & = & \begin{array}{rcl}\exists\mathbf{f}_2\,((\mathbf{f}_2 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_2,\mathbf{p})) & \rightarrow & \Psi(\mathbf{f},\mathbf{p}) \ \wedge\\ \neg\exists\mathbf{f}_2\,((\mathbf{f}_2 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_2,\mathbf{p})) & \rightarrow & \psi(\mathbf{f},\mathbf{p}) \text{ where}\end{array}\end{array}$$

$$\begin{array}{rcl}\psi(\mathbf{f},\mathbf{p}) & = & \begin{array}{l}\chi_{P,1}(\mathbf{f}, c, \mathbf{p})\wedge\\ \Upsilon_{P,1}(\mathbf{f}, c, t, \mathbf{p}) \wedge \Upsilon_{P,1}(\mathbf{f}, c, m, \mathbf{p}) \ \wedge\\ \forall x\,((x \neq c) \rightarrow \chi_{P,3}(\mathbf{f}, x, \mathbf{p})) \ \wedge\\ \forall x, y \neg\,((x = c \wedge y = t) \bigvee (x = c \wedge y = m))\\ \quad \rightarrow \Upsilon_{P,3}(\mathbf{f}, x, y, \mathbf{p}) \ \wedge\\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p})\end{array}\end{array}$$

Once again as reasoned earlier, if there is some $\mathbf{f}_{\xi_2}$ for which $\Phi_G^{\xi_2}(\mathbf{f}_{\xi_2}, \mathbf{p})$ is true, there is a unique $\mathbf{f}^*_{\xi_2}$ such that $\Phi_G^{\xi_2}(\mathbf{f}^*_{\xi_2}, \mathbf{p})$ and $\psi_{\xi_2}(\mathbf{f}^*_{\xi_2}, \mathbf{p})$ are true. This $\mathbf{f}^*_{\xi_2}$ gives the particular c node that is chosen for the graph transformation as described above in (2) and $\psi_{\xi_2}(\mathbf{f}^*_{\xi_2}, \mathbf{p})$ gives the sub-actions to be taken on all nodes and edges.

We can now write the other operations in a similar manner. In all of them, $\Phi_A^{\xi}(\mathbf{f}_\xi, \mathbf{p})$ is of the form

$$\begin{array}{rcl}\exists\mathbf{f}\,((\mathbf{f} \neq \mathbf{f}_\xi) \wedge \psi_\xi(\mathbf{f},\mathbf{p})) & \rightarrow & \Psi_\xi(\mathbf{f}_\xi,\mathbf{p}) \ \wedge\\ \neg\exists\mathbf{f}\,((\mathbf{f} \neq \mathbf{f}_\xi) \wedge \psi_\xi(\mathbf{f},\mathbf{p})) & \rightarrow & \psi_\xi(\mathbf{f}_\xi,\mathbf{p})\end{array}$$

Hence in the following only Φ_G^{ξ} and ψ_ξ have been mentioned. For better readability, we shall omit the 's' subsript for e.g. instead of $\mathcal{M}_s$ we shall simply write $\mathcal{M}$ and similarly for others. Also instead of $\chi_{P,i}$ we shall simply write χ_i and for $\Upsilon_{P,i}$ we shall write Υ_i.

Borrow(b, m):

Construct $O_{Borrow} = (\{\xi_1, \xi_2\}, \mathbf{p})$ where $\mathbf{p} = (b, m)$ as follows:

- $\xi_1 = (\Phi_G, \Phi_A), \mathbf{f} = (s, l)$.

$$\Phi_G(\mathbf{f},\mathbf{p}) = \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{M}(m) \wedge \mathcal{S}(s) \wedge \mathcal{L}(l) \\ N(b) \wedge N(m) \wedge N(s) \wedge \neg N(l) \wedge E(b,s) \end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l} \chi_1(\mathbf{f}, l, \mathbf{p}) \wedge \\ \Upsilon_1(\mathbf{f}, b, l, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, l, m, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, b, s, \mathbf{p}) \wedge \\ \forall x\,((x \neq l) \rightarrow \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg\,((x = b \wedge y = s) \bigvee (x = b \wedge y = l) \bigvee (x = l \wedge y = m)) \\ \quad \rightarrow \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

- $\xi_2 = (\Phi_G, \Phi_A), \mathbf{f} = (c, l)$.

$$\Phi_G(\mathbf{f},\mathbf{p}) = \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{M}(m) \wedge \mathcal{C}(c) \wedge \mathcal{L}(l) \\ N(b) \wedge N(m) \wedge N(c) \wedge \neg N(l) \wedge E(b,c) \wedge E(c,m) \end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l} \chi_1(\mathbf{f}, l, \mathbf{p}) \wedge \chi_2(\mathbf{f}, c, \mathbf{p}) \wedge \\ \Upsilon_2(\mathbf{f}, b, c, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, c, m, \mathbf{p}) \wedge \\ \Upsilon_1(\mathbf{f}, b, l, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, l, m, \mathbf{p}) \wedge \\ \forall x\,((x \neq l \wedge x \neq c) \rightarrow \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg((x = b \wedge y = c) \bigvee (x = c \wedge y = m) \bigvee \\ \quad (x = b \wedge y = l) \bigvee (x = l \wedge y = m)) \rightarrow \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

Return(b, m):

Construct $O_{Return} = (\{\xi_1, \xi_2, \xi_3\}, \mathbf{p})$ where $\mathbf{p} = (b, m)$ as follows:

- $\xi_1 = (\Phi_G, \Phi_A), \mathbf{f} = (s, l)$.

$$\Phi_G(\mathbf{f},\mathbf{p}) = \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{M}(m) \wedge \mathcal{S}(s) \wedge \mathcal{L}(l) \\ N(b) \wedge N(m) \wedge N(s) \wedge N(l) \wedge \\ E(b,l) \wedge E(l,m) \wedge \neg E(b,s) \\ \neg \exists c, t(\mathcal{C}(c) \wedge \mathcal{T}(t) \wedge N(c) \wedge N(t) \wedge E(c,t) \wedge E(b,t)) \end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l} \chi_2(\mathbf{f}, l, \mathbf{p}) \wedge \\ \Upsilon_2(\mathbf{f}, b, l, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, l, m, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, b, s, \mathbf{p}) \wedge \\ \forall x\,((x \neq l) \rightarrow \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg\,((x = b \wedge y = s) \bigvee (x = b \wedge y = l) \bigvee (x = l \wedge y = m)) \\ \quad \rightarrow \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

- $\xi_2 = (\Phi_G, \Phi_A), \mathbf{f} = (s, l)$.

$$\Phi_G(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{M}(m) \wedge \mathcal{S}(s) \wedge \mathcal{L}(l) \\ N(b) \wedge N(m) \wedge N(l) \wedge \neg N(s) \\ \neg\exists s_1(\mathcal{S}(s_1) \wedge N(s_1)) \wedge \\ E(b,l) \wedge E(l,m) \wedge \neg E(b,s) \\ \neg\exists c,t(\mathcal{C}(c) \wedge \mathcal{T}(t) \wedge N(c) \wedge N(t) \wedge E(c,t) \wedge E(b,t)) \end{array}$$

$$\psi(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \chi_2(\mathbf{f}, l, \mathbf{p}) \wedge \chi_1(\mathbf{f}, s, \mathbf{p}) \\ \Upsilon_2(\mathbf{f}, b, l, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, l, m, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, b, s, \mathbf{p}) \wedge \\ \forall x\, ((x \neq l) \to \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg\, ((x = b \wedge y = s) \bigvee (x = b \wedge y = l) \bigvee (x = l \wedge y = m)) \\ \quad \to \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

- $\xi_3 = (\Phi_G, \Phi_A), \mathbf{f} = (c, l, t)$.

$$\Phi_G(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{M}(m) \wedge \mathcal{C}(c) \wedge \mathcal{L}(l) \wedge \mathcal{T}(t) \\ N(b) \wedge N(m) \wedge N(c) \wedge N(l) \wedge N(t) \\ E(b,t) \wedge E(c,t) \wedge E(b,l) \wedge E(l,m) \end{array}$$

$$\psi(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \chi_2(\mathbf{f}, l, \mathbf{p}) \wedge \\ \Upsilon_1(\mathbf{f}, b, c, \mathbf{p}) \wedge \\ \Upsilon_2(\mathbf{f}, b, l, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, l, m, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, c, t, \mathbf{p}) \wedge \\ \forall x\, ((x \neq l) \to \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg((x = b \wedge y = l) \bigvee (x = l \wedge y = m) \bigvee \\ \quad (x = b \wedge y = c) \bigvee (x = c \wedge y = t)) \to \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

Addbook(b, t, s):

Construct $O_{\mathit{Addbook}} = (\{\xi_1, \xi_2, \xi_3, \xi_4, \xi_5\}, \mathbf{p})$ where $\mathbf{p} = (b, t, s)$ as follows:

- $\xi_1 = (\Phi_G, \Phi_A), \mathbf{f} = (d)$.

$$\Phi_G(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \mathcal{B}(b) \wedge \mathcal{T}(t) \wedge \mathcal{S}(s) \wedge \neg N(b) \wedge N(t) \wedge N(s) \\ \neg\exists c(\mathcal{C}(c) \wedge N(c) \wedge E(c,t)) \end{array}$$

$$\psi(\mathbf{f}, \mathbf{p}) \;=\; \begin{array}{l} \chi_1(\mathbf{f}, b, \mathbf{p}) \wedge \\ \Upsilon_1(\mathbf{f}, b, t, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, b, s, \mathbf{p}) \wedge \\ \forall x\, ((x \neq b) \to \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge \\ \forall x, y \neg\, ((x = b \wedge y = t) \vee (x = b \wedge y = s)) \to \; \Upsilon_3(\mathbf{f}, x, y, \mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p}) \end{array}$$

- $\xi_2 = (\Phi_G, \Phi_A), \mathbf{f} = (d)$.

$$\begin{array}{rcl}\Phi_G(\mathbf{f},\mathbf{p}) & = & \mathcal{B}(b) \wedge \mathcal{T}(t) \wedge \mathcal{S}(s) \wedge \neg N(b) \wedge \neg N(t) \wedge N(s) \\ & & \neg\exists c(\mathcal{C}(c) \wedge N(c) \wedge E(c,t))\end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l}\chi_1(\mathbf{f},b,\mathbf{p}) \wedge \chi_1(\mathbf{f},t,\mathbf{p})\wedge \\ \Upsilon_1(\mathbf{f},b,t,\mathbf{p}) \wedge \Upsilon_1(\mathbf{f},b,s,\mathbf{p})\wedge \\ \forall x\,((x \neq b \wedge x \neq t) \to \chi_3(\mathbf{f},x,\mathbf{p}))\wedge \\ \forall x,y\neg\,((x = b \wedge y = t) \vee (x = b \wedge y = s)) \to\ \Upsilon_3(\mathbf{f},x,y,\mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f},x,\mathbf{p})\end{array}$$

- $\xi_3 = (\Phi_G, \Phi_A), \mathbf{f} = (d)$.

$$\begin{array}{rcl}\Phi_G(\mathbf{f},\mathbf{p}) & = & \mathcal{B}(b) \wedge \mathcal{T}(t) \wedge \mathcal{S}(s) \wedge \neg N(b) \wedge N(t) \wedge \neg N(s) \\ & & \neg\exists c(\mathcal{C}(c) \wedge N(c) \wedge E(c,t))\end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l}\chi_1(\mathbf{f},b,\mathbf{p}) \wedge \chi_1(\mathbf{f},s,\mathbf{p})\wedge \\ \Upsilon_1(\mathbf{f},b,t,\mathbf{p}) \wedge \Upsilon_1(\mathbf{f},b,s,\mathbf{p})\wedge \\ \forall x\,((x \neq b \wedge x \neq s) \to \chi_3(\mathbf{f},x,\mathbf{p}))\wedge \\ \forall x,y\neg\,((x = b \wedge y = t) \vee (x = b \wedge y = s)) \to\ \Upsilon_3(\mathbf{f},x,y,\mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f},x,\mathbf{p})\end{array}$$

- $\xi_4 = (\Phi_G, \Phi_A), \mathbf{f} = (d)$.

$$\begin{array}{rcl}\Phi_G(\mathbf{f},\mathbf{p}) & = & \mathcal{B}(b) \wedge \mathcal{T}(t) \wedge \mathcal{S}(s) \wedge \neg N(b) \wedge \neg N(t) \wedge \neg N(s) \\ & & \neg\exists c(\mathcal{C}(c) \wedge N(c) \wedge E(c,t))\end{array}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l}\chi_1(\mathbf{f},b,\mathbf{p}) \wedge \chi_1(\mathbf{f},t,\mathbf{p}) \wedge \chi_1(\mathbf{f},s,\mathbf{p})\wedge \\ \Upsilon_1(\mathbf{f},b,t,\mathbf{p}) \wedge \Upsilon_1(\mathbf{f},b,s,\mathbf{p})\wedge \\ \forall x\,((x \neq b \wedge x \neq s \wedge x \neq t) \to \chi_3(\mathbf{f},x,\mathbf{p}))\wedge \\ \forall x,y\neg\,((x = b \wedge y = t) \vee (x = b \wedge y = s)) \to\ \Upsilon_3(\mathbf{f},x,y,\mathbf{p}) \wedge \\ \forall x \bigwedge_{i=1}^{i=6} \chi_{N,T_i,3}(\mathbf{f},x,\mathbf{p})\end{array}$$

- $\xi_5 = (\Phi_G, \Phi_A), \mathbf{f} = (c)$.

$$\begin{aligned}\Phi_G(\mathbf{f},\mathbf{p}) &= \mathcal{B}(b) \wedge \mathcal{T}(t) \wedge \neg N(b) \wedge N(t)\\ &\quad \mathcal{C}(c) \wedge N(c) \wedge E(c,t)\end{aligned}$$

$$\psi(\mathbf{f},\mathbf{p}) = \begin{array}{l}\chi_1(\mathbf{f}, b, \mathbf{p})\wedge\\ \Upsilon_1(\mathbf{f}, b, t, \mathbf{p}) \wedge \Upsilon_1(\mathbf{f}, b, c, \mathbf{p}) \wedge \Upsilon_2(\mathbf{f}, c, t, \mathbf{p})\wedge\\ \forall x\,((x \neq b) \rightarrow \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge\\ \forall x, y \neg\,((x = b \wedge y = t) \vee (x = b \wedge y = c) \vee (x = c \wedge y = t))\\ \quad \rightarrow\ \Upsilon_3(\mathbf{f}, x, y, \mathbf{p})\ \wedge\\ \forall x \bigwedge_{i=1}^{i=6} \chi_{T_i,3}(\mathbf{f}, x, \mathbf{p})\end{array}$$

<u>*AddMember(m)*</u>:

Construct $O_{Addmember} = (\{\xi_1\}, \mathbf{p})$ where $\mathbf{p} = (m)$ as follows:

- $\xi_1 = (\Phi_G, \Phi_A), \mathbf{f} = (d)$.

$$\begin{aligned}\Phi_G(\mathbf{f},\mathbf{p}) &= \mathcal{M}(m) \wedge \neg N(m)\\ \psi(\mathbf{f},\mathbf{p}) &= \begin{array}{l}\chi_1(\mathbf{f}, m, \mathbf{p})\wedge\\ \forall x\,((x \neq m) \rightarrow \chi_3(\mathbf{f}, x, \mathbf{p})) \wedge\\ \forall x, y\ \Upsilon_3(\mathbf{f}, x, y, \mathbf{p})\ \wedge\\ \forall x \bigwedge_{i=1}^{i=6} \chi_{N,T_i,3}(\mathbf{f}, x, \mathbf{p})\end{array}\end{aligned}$$

Having written the operations, we can write the GTS, $S_{Library}$ for the library system as $S_{Library} = (\Delta_{Library}, \rho)$, where

$$\Delta_{Library} = \{O_{Addbook}, O_{Addmember}, O_{Borrow}, O_{Return}, O_{Reserve}\}$$

We now make the following observations about $S_{Library}$. For each operation $O = (\Omega, \mathbf{p}) \in \Delta_{Library}$, for each $\xi \in \Omega$ we have the following:

1. $\Phi_A(\mathbf{f},\mathbf{p})$ is of the form
$$\begin{array}{rcl}\exists \mathbf{f}_1\,((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p})) & \rightarrow & \Psi(\mathbf{f},\mathbf{p})\ \bigwedge\\ \neg\exists \mathbf{f}_1\,((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p})) & \rightarrow & \psi(\mathbf{f},\mathbf{p})\end{array}$$

2. $\psi(\mathbf{f},\mathbf{p}) \rightarrow \neg\Psi(\mathbf{f},\mathbf{p})$

3. $\psi(\mathbf{f},\mathbf{p})$ is a $\forall^*$ formula in prenex normal form. By **Claim 3** then, given any σ_A^O-structure M with universe A such that $M \models \psi(\mathbf{f}_1, \mathbf{p}_1)$ where $\mathbf{f}_1 \in A^{|\mathbf{f}|}, \mathbf{p}_1 \in A^{|\mathbf{p}|}$, there exists $A_1 \subseteq A$ such that

(a) $\mathbf{f}_1 \in A_1^{|\mathbf{f}|}, \mathbf{p}_1 \in A_1^{|\mathbf{p}|}$

(b) $|A_1| \leq ||\mathbf{f}_1|| + |\mathbf{p}_1|$ (hence A_1 is of bounded size)

(c) $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 is such that $M_2 \models \psi(\mathbf{f}_1, \mathbf{p}_1)$.

4. $\Phi_G(\mathbf{f}, \mathbf{p})$ has the following properties:

   (a) It is a $\forall^*$ formula in prenex normal form.

   (b) No occurence of each of $x = y$ and $E_s(x, y)$ is such that one of the variables is quantified and the other is in $\mathbf{f}$. Thus both x and y are such that either (a) each is quantified or is in $\mathbf{p}$ or (b) each is free (i.e. belongs to $\mathbf{f}$ or $\mathbf{p}$).

   We later show that for Φ_G of the form stated, given a σ_s-structure M with universe A and any $\mathbf{p}_1 \in A^{|\mathbf{p}|}$, there exists $A_1 \subseteq A$ such that

   (a) $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$

   (b) A_1 is of bounded size

   (c) $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 is such that $M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1), \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$.
   In particular then, if $M \models \forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}_1)$, then
   $M_2 \models \forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}_1)$

We now prove the following result for a generalisation of the particular example of the Library system considered above.

We will require the following definition. Let $Sig(\varphi)$ be the set of all predicate and constant symbols occuring in a FOL formula (with equality) φ.

Definition 9.1 (EBS property) *A formula $\varphi(\boldsymbol{x})$ is said to satisfy the* ***Extensible Bounded Submodel*** *property with respect to $\sigma \subseteq \Sigma$ (denoted as $\varphi(\boldsymbol{x}) \in \boldsymbol{EBS}_\Sigma(\sigma)$) if $Sig(\varphi) = \Sigma$ and there exists a cardinal $\mathcal{B}$ (dependent on φ and σ in general) such that for every $\Sigma-$structure M having universe A with $|A| \geq \mathcal{B}$ and for every $\boldsymbol{a} \in A^{|\boldsymbol{x}|}$, if $M \models \varphi(\boldsymbol{a})$ then there exists $A_1 \subseteq A$ such that*

1. *$\boldsymbol{a} \in A_1^{|\boldsymbol{x}|}$*

2. *$|A_1| \leq \mathcal{B}$*

3. *For all A_2 such that $A_1 \subseteq A_2 \subseteq A$, if M_2' is the substructure of M generated by A_2, then there exists a $\Sigma-$structure M_2 such that $M_2|_\sigma = M_2'|_\sigma$ and $M_2 \models \varphi(\boldsymbol{a})$.*

Note that in the special case that $\Sigma = \sigma$, the condition (3) above reduces to
$\forall A_2$ such that $A_1 \subseteq A_2 \subseteq A$, if M_2 is the substructure of M generated by A_2, then $M_2 \models \psi(\boldsymbol{a})$.

9.1.2 Generalization

<u>Result 1</u>:

Consider a GTS $S = (\Delta, \rho)$ in which for every operation $O = (\Omega, \mathbf{p}) \in \Delta$, each GA pair $\xi \in \Omega$ satisfies the following properties:

1. $\Phi_A(\mathbf{f}, \mathbf{p})$ is of the form
$$\begin{array}{rcl} \exists \mathbf{f}_1 \left((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p}) \right) & \rightarrow & \Psi(\mathbf{f}, \mathbf{p}) \ \bigwedge \\ \neg \exists \mathbf{f}_1 \left((\mathbf{f}_1 \neq \mathbf{f}) \wedge \psi(\mathbf{f}_1, \mathbf{p}) \right) & \rightarrow & \psi(\mathbf{f}, \mathbf{p}) \end{array}$$
where $\psi(\mathbf{f}, \mathbf{p})$ is a formula over σ_A^O satisfying the following
   (a) $\psi(\mathbf{f}, \mathbf{p}) \rightarrow \neg \Psi(\mathbf{f}, \mathbf{p})$
   (b) $\psi(\mathbf{f}, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma_A^O$.
2. $\Phi_G(\mathbf{f}, \mathbf{p})$ is such that
   (a) $\Phi_G(\mathbf{f}, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma_s$.
   (b) $\forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma_s$.

Then $T^O_{ss'}(\mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma^O_{ss'}$.

<u>Proof:</u>

Firstly we prove the following:

<u>Claim 4</u>:

For each $\xi \in \Omega$, either

1. $M \models \neg \Phi_G(\mathbf{f}_1, \mathbf{p}_1)$ (and hence $M \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$) $\forall \mathbf{f}_1 \in A^{|\mathbf{f}|}$ or
2. There is a *unique* $\mathbf{f}^*_\xi$ such that $M \models \left(\Phi_G(\mathbf{f}^*_\xi, \mathbf{p}_1) \wedge \psi(\mathbf{f}^*_\xi, \mathbf{p}_1) \right)$ and $\forall \mathbf{f}_1 \neq \mathbf{f}^*_\xi, M \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$

<u>Proof:</u>

Let $S = \left\{ \mathbf{f}_1 \in A^{|\mathbf{f}|} | M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \right\}$.

For any $\mathbf{f}_1 \in A^{|\mathbf{f}|}$, we have

$$\begin{array}{lll} M & \models & \neg \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \\ & \models & \Psi(\mathbf{f}_1, \mathbf{p}_1) \qquad (\because M \models T^O_{ss'}(\mathbf{p}_1)) \\ & \models & \neg \psi(\mathbf{f}_1, \mathbf{p}_1) \qquad \textit{(by 1(a) in the statement of the Result)} \end{array}$$

1. If $S = \{\}$, then for each $\mathbf{f}_1 \in A^{|\mathbf{f}|}$, $M \models \neg\Phi_G(\mathbf{f}_1, \mathbf{p}_1)$ and thus from above, $M \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$

2. Else suppose that $M \models \neg\psi(\mathbf{f}_1, \mathbf{p}_1), \forall \mathbf{f}_1 \in S$. (For $\mathbf{f}_1 \notin S, M \models \neg\psi(\mathbf{f}_1, \mathbf{p}_1)$ as shown above.) Then $\forall \mathbf{f}_1 \in S$,

$$\begin{array}{rll} M & \models & \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \\ & \models & \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \qquad\qquad (\because M \models T^O_{ss'}(\mathbf{p}_1)) \\ & \models & \neg\exists \mathbf{f}\,((\mathbf{f} \neq \mathbf{f}_1) \wedge \psi(\mathbf{f}, \mathbf{p}_1)) \rightarrow \psi(\mathbf{f}_1, \mathbf{p}_1) \\ & \models & \psi(\mathbf{f}_1, \mathbf{p}_1) \end{array}$$

giving a contradiction. Hence there is atleast one $\mathbf{f}_1 \in S$ such that $M \models \psi(\mathbf{f}_1, \mathbf{p}_1)$.

Suppose $M \models \psi(\mathbf{f}_i, \mathbf{p}_1), i = 1, 2$ where $\mathbf{f}_1 \neq \mathbf{f}_2, \mathbf{f}_1, \mathbf{f}_2 \in S$. Now for $i, j \in \{1, 2\}, i \neq j$

$$\begin{array}{rll} M & \models & \Phi_G(\mathbf{f}_i, \mathbf{p}_1) \\ & \models & \Phi_A(\mathbf{f}_i, \mathbf{p}_1) \\ & \models & \exists \mathbf{f}\,((\mathbf{f} \neq \mathbf{f}_i) \wedge \psi(\mathbf{f}, \mathbf{p}_1)) \rightarrow \Psi(\mathbf{f}_i, \mathbf{p}_1) \\ & \models & ((\mathbf{f}_j \neq \mathbf{f}_i) \wedge \psi(\mathbf{f}_j, \mathbf{p}_1)) \\ & \models & \Psi(\mathbf{f}_i, \mathbf{p}_1) \qquad\qquad (\textit{by prec. 2 stmts}) \\ & \models & \neg\psi(\mathbf{f}_i, \mathbf{p}_1) \end{array}$$

giving a contradiction.
Thus there exists a *unique* $\mathbf{f}^*_\xi \in S$ such that $M \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$ and for $\mathbf{f}_1 \neq \mathbf{f}^*_\xi, \mathbf{f}_1 \in S, M \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$. For $\mathbf{f}_1 \notin S, M \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$ as already shown above.

Thus $M \models \Psi(\mathbf{f}_1, \mathbf{p}_1), \forall \mathbf{f}_1 \neq \mathbf{f}^*_\xi$.

□

Let
$$S = \left\{\xi | M \models \exists \mathbf{f} \Phi^\xi_G(\mathbf{f}, \mathbf{p}_1)\right\}$$
From **Claim 4**, for each $\xi \in S$, there is a *unique* $\mathbf{f}^*_\xi$ such that $M \models \Phi_G(\mathbf{f}^*_\xi, \mathbf{p}_1) \wedge \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$. Since ψ is a σ^O_A formula, from **Claim 1** we have, for each $\xi \in S$,

$$M|_{\sigma^O_A} \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$$

Then by 1(b) in the statement of the result, $\psi(\mathbf{f}_\xi, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma^O_A$. Then there exists $\mathcal{B}_{\xi,1}$ such that since $M \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$ there exists $A_{\xi,1} \subseteq A(\mathbf{f}^*_\xi \in A^{|\mathbf{f}_\xi|}_{\xi,1}, \mathbf{p}_1 \in A^{|\mathbf{p}|}_{\xi,1})$ satisfying the conditions of the property.
Similarly for each $\xi \in S$, by 2(a) of the statement of the result, $\Phi_G(\mathbf{f}_\xi, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$

where $\Sigma = \sigma_s$. Then there exists $\mathcal{B}_{\xi,2}$ such that since $M \models \Phi_G(\mathbf{f}^*_\xi, \mathbf{p}_1)$ there exists $A_{\xi,2} \subseteq A(\mathbf{f}^*_\xi \in A^{|\mathbf{f}_\xi|}_{\xi,2}, \mathbf{p}_1 \in A^{|\mathbf{p}|}_{\xi,2})$ satisfying the conditions of the property.
Similarly for each $\xi \notin S$, by 2(b) of the statement of the result, $\forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}) \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma_s$. Then there exists $\mathcal{B}_\xi$ such that since $M \models \forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}_1)$ there exists $A_\xi \subseteq A(\mathbf{p}_1 \in A^{|\mathbf{p}|}_\xi)$ satisfying the conditions of the property.
Then construct

$$A_1 = \bigcup_{\xi \in S} A_{\xi,1} \cup \bigcup_{\xi \in S} A_{\xi,2} \cup \bigcup_{\xi \notin S} A_\xi$$

Let A_2 be such that $A_1 \subseteq A_2 \subseteq A$. Let M_2 be the substructure of M generated by A_2. Note that $\mathbf{p}_1 \in A^{|\mathbf{p}|}_1$.

Consider $\xi \in S$.
From above,

$$M|_{\sigma^O_A} \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$$

Consider $M_2|_{\sigma^O_A}$ which is the substructure of $M|_{\sigma^O_A}$ generated by A_2. Since $A_{\xi,1} \subseteq A_2$, by the **EBS** property for $\psi(\mathbf{f}_\xi, \mathbf{p})$,

$$M_2|_{\sigma^O_A} \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1)$$

hence by **Claim 1**

$$M_2 \models \psi(\mathbf{f}^*_\xi, \mathbf{p}_1) \qquad \xi \in S \tag{9.1}$$

Similarly,

$$M_2 \models \Phi_G(\mathbf{f}^*_\xi, \mathbf{p}_1) \qquad \xi \in S \tag{9.2}$$
$$M_2 \models \forall \mathbf{f} \neg \Phi_G(\mathbf{f}, \mathbf{p}_1) \qquad \xi \notin S \tag{9.3}$$

For all $\xi \in \Omega, \Psi_\xi$ is a $\forall^*$ formula. Since $\mathbf{p}_1 \in A^{|\mathbf{p}|}_1$ and $A_1 \subseteq A_2$, $\mathbf{p}_1 \in A^{|\mathbf{p}|}_2$. Then from **Claim 3**, since M_2 is a substructure of M,

$$M \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \rightarrow M_2 \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A^{|\mathbf{f}|}_2 \tag{9.4}$$

Then from **Claim 4**, 9.1, 9.4, we get

$$M_2 \models \begin{cases} \Psi(\mathbf{f}_1, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}_2, \mathbf{f}_1 \neq \mathbf{f}^*_\xi,\ \xi \in S \\ \psi(\mathbf{f}_1, \mathbf{p}_1) & \mathbf{f}_1 = \mathbf{f}^*_\xi \\ \Psi(\mathbf{f}_1, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}_2,\ \xi \notin S \end{cases} \tag{9.5}$$

Consider an arbitrary $\xi \in \Omega$. We show that $M_2 \models T^\xi_{ss'}(\mathbf{p}_1)$.

- <u>Case 1:</u> $\xi \in S$
  Since $\Psi(.) \rightarrow \neg\psi(.)$, from 9.5 above, we have

$$M_2 \quad \models \quad \left\{ \begin{array}{ll} \exists \mathbf{f}(\mathbf{f} \neq \mathbf{f}_1 \wedge \psi(\mathbf{f}, \mathbf{p}_1)) \bigwedge \Psi(\mathbf{f}_1, \mathbf{p}_1) & \mathbf{f}_1 \in A_2^{|\mathbf{f}|}, \mathbf{f}_1 \neq \mathbf{f}^*_\xi \\ \neg\exists \mathbf{f}(\mathbf{f} \neq \mathbf{f}_1 \wedge \psi(\mathbf{f}, \mathbf{p}_1)) \bigwedge \psi(\mathbf{f}_1, \mathbf{p}_1) & \mathbf{f}_1 = \mathbf{f}^*_\xi \end{array} \right.$$

  Thus $M_2 \models \Phi_A(\mathbf{f}, \mathbf{p}_1) \ \ \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$.
  Now,
  If $M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1)$, then from above $M_2 \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1)$.
  If $M_2 \models \neg\Phi_G(\mathbf{f}_1, \mathbf{p}_1)$, then from 9.2, $\mathbf{f}_1 \neq \mathbf{f}^*_\xi$ and hence from 9.5, $M_2 \models \Psi(\mathbf{f}_1, \mathbf{p}_1)$.
  Thus

$$M_2 \models T^\xi_{ss'}(\mathbf{p}_1)$$

- <u>Case 2</u>: $\xi \notin S$
  Then from 9.3, $M_2 \models \forall \mathbf{f} \neg \ \Phi_G(\mathbf{f}, \mathbf{p}_1)$
  and from 9.5, $M_2 \models \forall \mathbf{f} \ \Psi(\mathbf{f}, \mathbf{p}_1)$
  and hence

$$M_2 \models T^\xi_{ss'}(\mathbf{p}_1)$$

Thus from both the above cases we have

$$M_2 \quad \models \quad \bigwedge_{\xi \in \Omega_o} T^\xi_{ss'}(\mathbf{p}_1) \qquad (9.6)$$

Let,

$$\begin{aligned} U_n(x, \mathbf{p}) &= \bigvee_{\xi \in \Omega} \exists \mathbf{f} \ \chi^\xi_n(\mathbf{f}, x, \mathbf{p}) & \lambda_n \in \rho_N \\ V_e(x, y, \mathbf{p}) &= \bigvee_{\xi \in \Omega} \exists \mathbf{f} \ \Upsilon^\xi_e(\mathbf{f}, x, y, \mathbf{p}) & \mu_e \in \rho_E \end{aligned}$$

We recall that $\Psi(\mathbf{f}, \mathbf{p}) = \left(\forall x \bigwedge_{\lambda_n \in \rho_N} \neg\chi_n(\mathbf{f}, x, \mathbf{p})\right) \wedge \left(\forall x, y \bigwedge_{\mu_e \in \rho_E} \neg\Upsilon_e(\mathbf{f}, x, y, \mathbf{p})\right)$.
Then from **Claim 4**, for $\lambda_n \in \rho_N, \mu_e \in \rho_E, x, y \in A$, we have

$$M \quad \models \quad \left\{ \begin{array}{ll} \neg\chi_n(\mathbf{f}_1, x, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}, \ \xi \notin S \\ \neg\Upsilon_e(\mathbf{f}_1, x, y, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}, \ \xi \notin S \\ \neg\chi_n(\mathbf{f}_1, x, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}, \mathbf{f}_1 \neq \mathbf{f}^*_\xi, \ \xi \in S \\ \neg\Upsilon_e(\mathbf{f}_1, x, y, \mathbf{p}_1) & \mathbf{f}_1 \in A^{|\mathbf{f}|}, \mathbf{f}_1 \neq \mathbf{f}^*_\xi, \ \xi \in S \end{array} \right.$$

Then, for $x, y \in A$, the following holds in M.

$$\begin{aligned} U_n(x, \mathbf{p}_1) &= \bigvee_{\xi \in S} \chi^\xi_n(\mathbf{f}^*_\xi, x, \mathbf{p}_1) & \lambda_n \in \rho_N \\ V_e(x, y, \mathbf{p}_1) &= \bigvee_{\xi \in S} \Upsilon^\xi_e(\mathbf{f}^*_\xi, x, y, \mathbf{p}_1) & \mu_e \in \rho_E \end{aligned}$$

We can similarly show using 9.5 that the above relation holds in M_2 as well. Thus in M and M_2, $U_n(x, \mathbf{p}_1)$ and $V_e(x, y, \mathbf{p}_1)$ are in fact quantifier-free formulae.
Consider the formulae $\chi_n(x, \mathbf{p}), \alpha_n(x, \mathbf{p})(\lambda_n \in \rho_N)$ and $\Upsilon_e(x, y, \mathbf{p}), \beta_e(x, y, \mathbf{p})(\mu_e \in \rho_E)$. These are quantifier free formulae. Since $M_2 \subseteq M$ by **Claim 2** and **Claim 1**, for $x, y \in A_2, \lambda_n \in \rho_N, \mu_e \in \rho_E$

$$\begin{aligned}
M_2 \models \chi_n(x, \mathbf{p}_1) &\leftrightarrow M \models \chi_n(x, \mathbf{p}_1) \\
M_2 \models \Upsilon_e(x, y, \mathbf{p}_1) &\leftrightarrow M \models \Upsilon_e(x, y, \mathbf{p}_1) \\
M_2 \models U_n(x, \mathbf{p}_1) &\leftrightarrow M \models U_n(x, \mathbf{p}_1) \\
M_2 \models V_e(x, y, \mathbf{p}_1) &\leftrightarrow M \models V_e(x, y, \mathbf{p}_1) \\
M_2 \models \alpha_n(x, \mathbf{p}_1) &\leftrightarrow M \models \alpha_n(x, \mathbf{p}_1) \\
M_2 \models \beta_e(x, y, \mathbf{p}_1) &\leftrightarrow M \models \beta_e(x, y, \mathbf{p}_1)
\end{aligned}$$

Using the above and the fact that since $M \models T^O_{ss'}(\mathbf{p}_1)$, $M \models Z^O(\mathbf{p}_1) \wedge \Gamma^O(\mathbf{p}_1)$, we have

$$M_2 \models Z^O(\mathbf{p}_1) \wedge \Gamma^O(\mathbf{p}_1) \tag{9.7}$$

From 9.5 and 9.6,

$$M_2 \models T^O_{ss'}(\mathbf{p}_1)$$

Note that $A_1 \subseteq A_2$ and $\mathbf{p}_1 \in A^{|\mathbf{p}|}$

$\square$

<u>Bound on $|A_1|$</u>:

$$\begin{aligned}
A_1 &= \bigcup_{\xi \in S}(A_{\xi,1} \cup A_{\xi,2}) \cup \bigcup_{\xi \notin S} A_\xi \\
|A_1| &\leq \Sigma_{\xi \in S}(|A_{\xi,1}| + |A_{\xi,2}|) + \Sigma_{\xi \notin S}|A_\xi| \\
&\leq \Sigma_{\xi \in S}(\mathcal{B}_{\xi,1} + \mathcal{B}_{\xi,2}) + \Sigma_{\xi \notin S}\mathcal{B}_\xi \\
&\leq \mathbf{max}_{S \subseteq \Omega} \{\Sigma_{\xi \in S}(\mathcal{B}_{\xi,1} + \mathcal{B}_{\xi,2}) + \Sigma_{\xi \notin S}\mathcal{B}_\xi\} \\
&= \mathcal{B}_O(say)
\end{aligned}$$

<u>**Corollary 1**</u>:

Consider a GTS $S = (\Delta, \rho)$ where each operation has the properties stated in the previous result. Then $T_{ss'} \in \mathbf{EBS}_\Sigma(\Sigma)$ where $\Sigma = \sigma_{ss'}$.

Proof:

$$T_{ss'} = \bigvee_{O=(\Omega,\mathbf{p})\in\Delta} \exists \mathbf{p} T^{O}_{ss'}(\mathbf{p})$$

Since $M \models T_{ss'}$, $M \models T^{O}_{ss'}(\mathbf{p}_1)$ for some $O \in \Delta$ and some $\mathbf{p}_1 \in A^{|\mathbf{p}|}$. By **Claim 1**, $M|_{\sigma^{O}_{ss'}} \models T^{O}_{ss'}(\mathbf{p}_1)$. Then by **Result 1**, since $T^{O}_{ss'}(\mathbf{p}) \in \mathbf{EBS}_{\Sigma}(\Sigma)$ where $\Sigma = \sigma^{O}_{ss'}$, there exists a $\mathcal{B}_O$ such that since $M|_{\sigma^{O}_{ss'}} \models T^{O}_{ss'}(\mathbf{p}_1)$ there exists $A_1 \subseteq A$ such that

1. $|A_1| \leq \mathcal{B}_O$
2. $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$
3. $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M'_2 of $M|_{\sigma^{O}_{ss'}}$ generated by A_2 is such that $M'_2 \models T^{O}_{ss'}(\mathbf{p}_1)$.

Then consider the substructure M_2 of M generated by A_2. CLearly $M_2|_{\sigma^{O}_{ss'}} = M'_2$. Since $M'_2 \models T^{O}_{ss'}(\mathbf{p}_1)$ by **Claim 1**, we have $M_2 \models T^{O}_{ss'}(\mathbf{p}_1)$ and hence $M_2 \models T_{ss'}$.

$|A_1| \leq \mathcal{B}_O \leq \mathbf{max}_{O\in\Delta}\mathcal{B}_O \ (= \mathcal{B})$

□

9.2 Class 2

9.2.1 An Example

Before considering the example, we need the following definition:

Let σ be a vocabulary. Let $\varphi(\mathbf{f})$ be a formula over σ. Then $\varphi(\mathbf{f})$ is said to be *positive unate* in a k-ary predicate $P \in \sigma$ if the following is true:

Let M be any σ-structure (with universe A) s.t. $M \models \varphi(\mathbf{f}_1), (\mathbf{f}_1 \in A^{|\mathbf{f}|})$. Then each σ-structure M_1 with the same universe A and which differs from M only in the value of $P(x_1, x_2, \ldots, x_k)$ being false in M and true in M_1, is such that $M_1 \models \varphi(\mathbf{f}_1)$.

Now consider a GTS $S = (\Delta, \rho)$, where for each operation $O = (\Omega, \mathbf{p}) \in \Delta$, each GA pair $\xi \in \Omega$, has the following properties:

1. $\Phi_G(\mathbf{f}, \mathbf{p})$ has the following properties:

   (a) It is a $\exists^*$ (or a $\forall^*$) formula in prenex normal form.

(b) No occurence of each of $x = y$, $E_s(x, y)$ and $T^i_{E,s}(x, y)(1 \le i \le \mathcal{N}_e)$ is such that one of the variables is quantified and the other is in $\mathbf{f}$. Thus both x and y are such that either (i) each is quantified or is in $\mathbf{p}$ or (ii) each is free (i.e. belongs to $\mathbf{f}$ or $\mathbf{p}$).

2. $\Phi_A(\mathbf{f}, \mathbf{p})$ is a $\forall^*$ formula in prenex normal form and is positive unate in the predicates of $\sigma_N^{all\ \xi} \cup \sigma_E^{all\ \xi}$ (i.e. $\left\{\chi_n^\xi | \lambda_n \in \rho_N, \xi \in \Omega\right\} \cup \left\{\Upsilon_e^\xi | \mu_e \in \rho_E, \xi \in \Omega\right\}$)

We now prove the following:

Φ_G is such that for any σ_s-structure M with universe A and for any $\mathbf{p}_1 \in A^{|\mathbf{p}|}$, there exists a subset $A_1 \subseteq A$ such that

1. A_1 is of bounded size
2. $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$
3. $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 is such that

$$M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

Proof:

Suppose $\Phi_G(\mathbf{f}, \mathbf{p})$ is an $\exists^*$ formula.(The case when it is a $\forall^*$ formula is similar) Then in disjunctive normal form, $\Phi_G(\mathbf{f}, \mathbf{p})$ can be written as

$$\exists \mathbf{y} \bigvee_{i=1}^{i=n} D_i(\mathbf{f}, \mathbf{y}, \mathbf{p})$$

where $D_1, D_2, \ldots, D_n$ are the disjuncts of the matrix of Φ_G. Each disjunct is a conjunction of atomic formulae and can be written as

$$D_i(\mathbf{f}, \mathbf{y}, \mathbf{p}) = C_i(\mathbf{f}, \mathbf{p}) \wedge B_i(\mathbf{y}, \mathbf{p})$$

where C_i is quantifier-free and has only the variables of $\mathbf{f}$ and $\mathbf{p}$ appearing in it and B_i is quantifier-free and has only the variables of $\mathbf{y}$ and $\mathbf{p}$ appearing in it.
(This follows from 1(b)(i) and 1(b)(ii) above.)
If $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ then

$$\Phi_G(\mathbf{f}, \mathbf{p}) = \exists y_1, y_2, \ldots, y_m \bigvee_{i=1}^{i=n} D_i(\mathbf{f}, y_1, y_2, \ldots, y_m, \mathbf{p})$$

As the $\exists$ distributes over disjunction,

$$\Phi_G(\mathbf{f}, \mathbf{p}) = \exists y_1, y_2, \ldots, y_{m-1} \bigvee_{i=1}^{i=n} \exists y_m D_i(\mathbf{f}, y_1, y_2, \ldots, y_m, \mathbf{p})$$

Continuing this way for the remaining y_js,

$$\begin{aligned}
\Phi_G(\mathbf{f},\mathbf{p}) &= \bigvee_{i=1}^{i=n} \exists \mathbf{y} D_i(\mathbf{f},\mathbf{y},\mathbf{p}) \\
&= \bigvee_{i=1}^{i=n} \exists \mathbf{y}\,(C_i(\mathbf{f},\mathbf{p}) \wedge B_i(\mathbf{y},\mathbf{p})) \\
&= \bigvee_{i=1}^{i=n} (C_i(\mathbf{f},\mathbf{p}) \wedge \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}))
\end{aligned}$$

Now given a σ_s-structure M with universe A and any $\mathbf{p}_1 \in A^{|\mathbf{p}|}$, let

$$S = \{i | 1 \leq i \leq n, M \models \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1)\}$$

By **Claim 3**, for each $i \in S$, there exists $A_{i,1} \subseteq A$ such that $\mathbf{p}_1 \in A_{i,1}^{|\mathbf{p}|}$ and $|A_{i,1}| \leq ||\mathbf{p}_1|| + m$ satisfying the properties given in the claim. Then construct

$$A_1 = \bigcup_{i \in S} A_{i,1}$$

Then $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$ and $|A_1| \leq \Sigma_{i \in S} |A_{i,1}| \leq n.(||\mathbf{p}|| + m))$. Thus A_1 is bounded. Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$ and let M_2 be the substructure of M generated by A_2.

1. Since $C_i (1 \leq i \leq n)$ is a quantifier-free formula and M_2 is a substructure of M, for any $\mathbf{f}_1 \in A_2^{|\mathbf{f}|}$, we have from **Claim 2**,

$$M_2 \models C_i(\mathbf{f}_1,\mathbf{p}_1) \leftrightarrow M \models C_i(\mathbf{f}_1,\mathbf{p}_1)$$

2. For $i \in S, A_{i,1} \subseteq A_2$. Then from **Claim 3**

$$M_2 \models \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1)$$

For $i \notin S$,

$$\begin{aligned}
M &\models \neg \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1) \\
&\models \forall \mathbf{y} \neg B_i(\mathbf{y},\mathbf{p}_1)
\end{aligned}$$

Then again from **Claim 3**, since $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$, we have

$$\begin{aligned}
M_2 &\models \forall \mathbf{y} \neg B_i(\mathbf{y},\mathbf{p}_1) \\
&\models \neg \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1)
\end{aligned}$$

Thus

$$M \models \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1) \leftrightarrow M_2 \models \exists \mathbf{y} B_i(\mathbf{y},\mathbf{p}_1) \qquad 1 \leq i \leq n$$

Then for any $\mathbf{f}_1 \in A_2^{|\mathbf{f}|}$, since $\Phi_G(\mathbf{f}_1, \mathbf{p}_1)$ is a boolean combination of $C_i(\mathbf{f}_1, \mathbf{p}_1)$ and $\exists \mathbf{y} B_i(\mathbf{y}, \mathbf{p}_1)$ $(1 \le i, j \le n)$,

$$M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

If Φ_G is a $\forall^*$ formula, then writing its matrix in conjunctive normal form and proceeding similarly, we can prove the above equivalence.

$\square$

Suppose now that for a σ_A^O−structure M with universe A, and for $\mathbf{p}_1 \in A^{|\mathbf{p}|}$,

$$M \models \bigvee_{\xi \in \Omega} \exists \mathbf{f} \left(\Phi_G(\mathbf{f}, \mathbf{p}_1) \wedge \Phi_A(\mathbf{f}, \mathbf{p}_1)\right)$$

Then there is $\xi^* \in \Omega$ and $\mathbf{f}_{\xi^*} \in A^{|\mathbf{f}|}$ such that

$$M \models \left(\Phi_G^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1) \wedge \Phi_A^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1)\right)$$

Construct A_1 as shown above for $\Phi_G^{\xi^*}$. A_1 is of bounded size. $\mathbf{f}_{\xi^*} \in A_1^{|\mathbf{f}|}, \mathbf{p}_1 \in A_1^{|\mathbf{p}|}$. Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$. Let M_2 be the substructure of M generated by A_2.
Since for all $\xi \in \Omega, \Phi_A$ is a $\forall^*$ formula, from **Claim 3**, we have

$$M \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \;\rightarrow\; M_2 \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|} \tag{9.8}$$

Let $A_n \subseteq A_2 (\lambda_n \in \rho_N)$ and $A_e \subseteq A_2 \times A_2 (\mu_e \in \rho_E)$. Then consider M_2' which is exactly the same as M_2 except that $\chi_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$ is set to **True** $(x \in A_n, \lambda_n \in \rho_N)$ and $\Upsilon_e^{\xi^*}(\mathbf{f}_{\xi^*}, x, y, \mathbf{p}_1)$ is set to **True** $((x, y) \in A_e, \mu_e \in \rho_E)$.
For any $\xi \in \Omega$, since Φ_A is positive unate in the $\chi_n^{\xi^*}(\lambda_n \in \rho_N)$ and $\Upsilon_e^{\xi^*}(\mu_e \in \rho_E)$ predicates, we have

$$M_2 \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \;\rightarrow\; M_2' \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|} \tag{9.9}$$

From 9.8 and 9.9 above, for all $\xi \in \Omega$

$$M \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \;\rightarrow\; M_2' \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

We now prove the following result for a generalisation of the particular example considered above.

9.2.2 Generalisation

Result 2:

Consider a GTS $S = (\Delta, \rho)$ in which every operation $O = (\Omega, \mathbf{p})$ satisfies the following properties:

1. For any GA pair $\xi = (\Phi_G, \Phi_A) \in \Omega$, Φ_G is such that for any σ_s-structure M with universe A and for any $\mathbf{p}_1 \in A^{|\mathbf{p}|}$, there exists a subset $A_\xi \subseteq A$ such that
   (a) A_ξ is of bounded size
   (b) $\mathbf{p}_1 \in A_\xi^{|\mathbf{p}|}$
   (c) $\forall A_2, A_\xi \subseteq A_2 \subseteq A$, the substructure M_2 of M generated by A_2 is such that
$$M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$
2. For any σ_A^O-structure M, for any $\mathbf{p}_1 \in A^{|\mathbf{p}|}$, if $M \models \bigvee_{\xi\in\Omega} \exists \mathbf{f}\, (\Phi_G(\mathbf{f}, \mathbf{p}_1) \wedge \Phi_A(\mathbf{f}, \mathbf{p}_1))$, then there exists $\xi^* \in \Omega, \mathbf{f}_{\xi^*} \in A^{|\mathbf{f}|}$ such that $M \models \left(\Phi_G^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1) \wedge \Phi_A^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1)\right)$ and there exists $A_1 \subseteq A$ such that
   (a) A_1 is of bounded size
   (b) $\mathbf{f}_{\xi^*} \in A_1^{|\mathbf{f}|}$ and $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$
   (c) $\forall A_2, A_1 \subseteq A_2 \subseteq A$, the substructure M_2' of M generated by A_2 satisfies the following:

   For any subset $A_n \subseteq A_2 (\lambda_n \in \rho_N)$ and any subset $A_e \subseteq A_2 \times A_2 (\mu_e \in \rho_E)$, the σ_A^O-structure M_2 obtained from M_2' by setting $\chi_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$ to **True** ($x \in A_n, \lambda_n \in \rho_N$)and $\Upsilon_e^{\xi^*}(\mathbf{f}_{\xi^*}, x, y, \mathbf{p}_1)$ to **True** ($(x, y) \in A_e, \mu_e \in \rho_E$) is such that
$$M \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \rightarrow M_2 \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}, \xi \in \Omega$$

Then $T_{ss'}^O(\mathbf{p}) \in \mathbf{EBS}_\Sigma(\sigma)$ where $\Sigma = \sigma_{ss'}^O$ and $\sigma = \sigma_s \cup \sigma_{s'}$.

Proof:

We are given $\sigma_{ss'}^O$-structure M with universe A such that $M \models T_{ss'}^O(\mathbf{p}_1)$ for $\mathbf{p}_1 \in A^{|\mathbf{p}|}$. Consider $M|_{\sigma_s}$. Then for each $\xi \in \Omega$, for Φ_G, there exists a subset $A_\xi \subseteq A$ such that $\mathbf{p}_1 \in A_\xi^{|\mathbf{p}|}, A_\xi$ is bounded in size ($|A_\xi| \leq \mathcal{B}_\xi$) and satisfies the properties stated above.

1. If $M \models \neg \bigvee_{\xi\in\Omega} \exists \mathbf{f} \Phi_G^\xi(\mathbf{f}, \mathbf{p}_1)$, then construct
$$A_1 = \bigcup_{\xi\in\Omega} A_\xi$$

Note that $A_1 \subseteq A$, $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$.
$|A_1| \leq \Sigma_{\xi\in\Omega}|A_\xi| \leq \Sigma_{\xi\in\Omega}\mathcal{B}_\xi$.
Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$. Construct M_2 as the substructure of M generated by A_2. Now, since $M \models T^O_{ss'}(\mathbf{p}_1)$ and $M \models \neg \bigvee_{\xi\in\Omega} \exists \mathbf{f}\Phi^\xi_G(\mathbf{f}, \mathbf{p}_1)$,

$$M \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A^{|\mathbf{f}|}, \xi \in \Omega \tag{9.10}$$

Since Ψ is a $\forall^*$ formula, $\mathbf{p}_1 \in A_2^{|\mathbf{p}|}$ and $M_2 \subseteq M$, by **Claim 2**,

$$M_2 \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|}, \xi \in \Omega \tag{9.11}$$

Further since for each $\xi \in \Omega, A_\xi \subseteq A_2$ and $M_2|_{\sigma_s} \subseteq M|_{\sigma_s}$, from assumption,

$$M_2|_{\sigma_s} \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M|_{\sigma_s} \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

Then by **Claim 1**,

$$M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

Since $M \models \neg \bigvee_{\xi\in\Omega} \exists \mathbf{f}\Phi_G(\mathbf{f}, \mathbf{p}_1)$,

$$M_2 \models \neg\Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|}, \xi \in \Omega \tag{9.12}$$

From 9.11 and 9.12, we can see that

$$M_2 \models T^\xi_{ss'}(\mathbf{p}_1) \qquad \xi \in \Omega$$

Let

$$\begin{aligned} U_n(x, \mathbf{p}) &= \bigvee_{\xi\in\Omega} \exists \mathbf{f}\chi^\xi_n(\mathbf{f}, x, \mathbf{p}) \qquad \lambda_n \in \rho_N \\ V_e(x, y, \mathbf{p}) &= \bigvee_{\xi\in\Omega} \exists \mathbf{f}\Upsilon^\xi_e(\mathbf{f}, x, y, \mathbf{p}) \qquad \mu_e \in \rho_E \end{aligned}$$

Then from 9.10 and 9.11 we can also see that

$$\begin{aligned} M_2 \models U_n(x, \mathbf{p}_1) &\leftrightarrow M \models U_n(x, \mathbf{p}_1) \qquad x \in A_2, \lambda_n \in \rho_N \\ M_2 \models V_e(x, y, \mathbf{p}_1) &\leftrightarrow M \models V_e(x, y, \mathbf{p}_1) \qquad x, y \in A_2, \mu_e \in \rho_E \end{aligned}$$

We shall use these facts later.

2. If $M \models \bigvee_{\xi\in\Omega} \exists \mathbf{f}\Phi_G^{\xi}(\mathbf{f}, \mathbf{p}_1)$, then since $M \models T_{ss'}^{O}(\mathbf{p}_1)$

$$M \models \bigwedge_{\xi\in\Omega} \forall \mathbf{f} \left(\Phi_G^{\xi}(\mathbf{f}, \mathbf{p}_1) \rightarrow \Phi_A^{\xi}(\mathbf{f}, \mathbf{p}_1)\right)$$

and therefore $M \models \bigvee_{\xi\in\Omega} \exists \mathbf{f} \left(\Phi_G^{\xi}(\mathbf{f}, \mathbf{p}_1) \wedge \Phi_A^{\xi}(\mathbf{f}, \mathbf{p}_1)\right)$.

Then by assumption, there exists $\xi^* \in \Omega$ and $\mathbf{f}_{\xi^*} \in A^{|\mathbf{f}|}$ such that $M \models \left(\Phi_G^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1) \wedge \Phi_A^{\xi^*}(\mathbf{f}_{\xi^*}, \mathbf{p}_1)\right)$ and there exists $A^*_{\xi^*} \subseteq A$ which is bounded in size ($|A^*_{\xi^*}| \leq \mathcal{B}^*_{\xi^*}$) satisfying the assumptions.

Then construct

$$A_1 = \bigcup_{\xi\in\Omega} A_\xi \cup A^*_{\xi^*}$$

Note that $A_1 \subseteq A$, $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$ and $|A_1| \leq (\Sigma_{\xi\in\Omega}|A_\xi| + |A^*_{\xi^*}|) \leq (\Sigma_{\xi\in\Omega}\mathcal{B}_\xi + \mathcal{B}^*_{\xi^*})$.
Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$. Construct the substructure M_2' of M generated by A_2.
Consider $x \in A_2$. For $\lambda_n \in \rho_N$, if $M_2' \models \chi_n^{\xi}(\mathbf{f}_1, x, \mathbf{p}_1)$ for some $\xi \in \Omega$ and some $\mathbf{f}_1 \in A_2^{|\mathbf{f}|}$, then $M_2' \models U_n(x, \mathbf{p}_1)$. Then as $M_2' \subseteq M$, $M \models \chi_n^{\xi}(\mathbf{f}_1, x, \mathbf{p}_1)$ and hence $M \models U_n(x, \mathbf{p}_1)$. However vice-versa might not be true. Like-wise for $\mu_e \in \rho_E$. Hence let

$$\begin{array}{rcll} A_n &=& \left\{x \in A_2 \;\middle|\; \begin{array}{ccl} M_2' & \models & \neg U_n(x, \mathbf{p}_1) \quad \text{but} \\ M & \models & U_n(x, \mathbf{p}_1) \end{array}\right\}, & \lambda_n \in \rho_N \\ A_e &=& \left\{(x,y) \in A_2 \times A_2 \;\middle|\; \begin{array}{ccl} M_2' & \models & \neg V_e(x, y, \mathbf{p}_1) \quad \text{but} \\ M & \models & V_e(x, y, \mathbf{p}_1) \end{array}\right\}, & \mu_e \in \rho_E \end{array}$$

Consider the σ_A^O-structure M_2 which is the same as M_2' except that $\chi_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$ is set to **True** ($x \in A_n, \lambda_n \in \rho_N$)and $\Upsilon_e^{\xi^*}(\mathbf{f}_{\xi^*}, x, y, \mathbf{p}_1)$ is set to **True** ($(x, y) \in A_e, \mu_e \in \rho_E$). By the assumption about Φ_A, for each $\xi \in \Omega$, we have

$$M \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \;\rightarrow\; M_2 \models \Phi_A(\mathbf{f}_1, \mathbf{p}_1) \quad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|} \tag{9.13}$$

Now $M_2|_{\sigma_s} = M_2'|_{\sigma_s} \subseteq M|_{\sigma_s}$. Since for each $\xi \in \Omega, A_\xi \subseteq A_2$, by assumptions about Φ_G,

$$M_2|_{\sigma_s} \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \leftrightarrow M_{\sigma_s} \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|}$$

By **Claim 1** then

$$M_2 \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \;\leftrightarrow\; M \models \Phi_G(\mathbf{f}_1, \mathbf{p}_1) \qquad \forall \mathbf{f}_1 \in A_2^{|\mathbf{f}|} \tag{9.14}$$

Since Ψ is a $\forall^*$ formula, $\mathbf{p}_1 \in A_2^{|\mathbf{p}|}$ and $M_2' \subseteq M$, by **Claim 2**,

$$M \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \;\rightarrow\; M_2' \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \qquad \mathbf{f}_1 \in A_2^{|\mathbf{f}|} \tag{9.15}$$

Since M_2 differs from M_2' only in the $\chi_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$ and the $\Upsilon_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$ values, we have for $\mathbf{f}_1 \in A_2^{|\mathbf{f}|}$,

$$M_2' \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \;\leftrightarrow\; M_2 \models \Psi(\mathbf{f}_1, \mathbf{p}_1) \tag{9.16}$$

From 9.13, 9.14, 9.15, 9.16, we have

$$M_2 \models T_{ss'}^{\xi}(\mathbf{p}_1) \qquad \xi \in \Omega$$

Now for any $x \in A_2$ and $\lambda_n \in \rho_N$

- If $x \in A_n$, then $M \models U_n(x, \mathbf{p}_1)$ but $M_2' \models \neg U_n(x, \mathbf{p}_1)$. Since $M_2 \models \chi_n^{\xi^*}(\mathbf{f}_{\xi^*}, x, \mathbf{p}_1)$, $M_2 \models U_n(x, \mathbf{p}_1)$.
- If $x \notin A_n$, then $M \models U_n(x, \mathbf{p}_1) \leftrightarrow M_2' \models U_n(x, \mathbf{p}_1)$. From the way M_2 was constructed, the $\chi_n^{\xi}(\mathbf{f}, z, \mathbf{p}_1)$ values were changed only for $\xi = \xi^*, \mathbf{f} = \mathbf{f}_{\xi^*}$ and $z \in A_n$. Since $x \notin A_n$, $M_2 \models U_n(x, \mathbf{p}_1) \leftrightarrow M_2' \models U_n(x, \mathbf{p}_1)$

From either case above,

$$M_2 \models U_n(x, \mathbf{p}_1) \leftrightarrow M \models U_n(x, \mathbf{p}_1) \qquad x \in A_2, \lambda_n \in \rho_N$$

Similarly,

$$M_2 \models V_e(x, y, \mathbf{p}_1) \leftrightarrow M \models V_e(x, y, \mathbf{p}_1) \qquad x, y \in A_2, \mu_e \in \rho_E$$

In both cases (1) and (2) above we have shown that,

$$\begin{aligned} M_2 \;&\models\; \bigwedge_{\xi \in \Omega} T_{ss'}^{\xi}(\mathbf{p}_1) && \qquad (9.17)\\ M_2 \models U_n(x, \mathbf{p}_1) \;&\leftrightarrow\; M \models U_n(x, \mathbf{p}_1) && \qquad x \in A_2, \lambda_n \in \rho_N \\ M_2 \models V_e(x, y, \mathbf{p}_1) \;&\leftrightarrow\; M \models V_e(x, y, \mathbf{p}_1) && \qquad x, y \in A_2, \mu_e \in \rho_E \end{aligned}$$

Consider the formulae $\chi_n(x, \mathbf{p}), \alpha_n(x, \mathbf{p})(\lambda_n \in \rho_N)$ and $\Upsilon_e(x, y, \mathbf{p}), \beta_e(x, y, \mathbf{p})(\mu_e \in \rho_E)$. These are quantifier free formulae. Now since

$$M_2|_{\sigma_s \cup \sigma_{s'} \cup \sigma_N \cup \sigma_E} = M_2'|_{\sigma_s \cup \sigma_{s'} \cup \sigma_N \cup \sigma_E} \subseteq M|_{\sigma_s \cup \sigma_{s'} \cup \sigma_N \cup \sigma_E}$$

applying **Claim 2** and subsequently **Claim 1**,

$$\begin{array}{rcll} M_2 \models \chi_n(x, \mathbf{p}_1) & \leftrightarrow & M \models \chi_n(x, \mathbf{p}_1) & x \in A_2, \lambda_n \in \rho_N \\ M_2 \models \Upsilon_e(x, y, \mathbf{p}_1) & \leftrightarrow & M \models \Upsilon_e(x, y, \mathbf{p}_1) & x, y \in A_2, \mu_e \in \rho_E \\ M_2 \models \alpha_n(x, \mathbf{p}_1) & \leftrightarrow & M \models \alpha_n(x, \mathbf{p}_1) & x \in A_2, \lambda_n \in \rho_N \\ M_2 \models \beta_e(x, y, \mathbf{p}_1) & \leftrightarrow & M \models \beta_e(x, y, \mathbf{p}_1) & x, y \in A_2, \mu_e \in \rho_E \end{array}$$

Using the above and the fact that since $M \models T^O_{ss'}(\mathbf{p}_1)$, $M \models Z^O(\mathbf{p}_1) \wedge \Gamma^O(\mathbf{p}_1)$, we have

$$M_2 \models Z^O(\mathbf{p}_1) \wedge \Gamma^O(\mathbf{p}_1) \tag{9.18}$$

From 9.17 and 9.18,

$$M_2 \models T^O_{ss'}(\mathbf{p}_1)$$

Note that $M_2|_{\sigma_s \cup \sigma_{s'}} \subseteq M|_{\sigma_s \cup \sigma_{s'}}$ and $A_1 \subseteq A_2 \subseteq A, \mathbf{p}_1 \in A_1^{|\mathbf{p}|}$.
$|A_1| \leq (\Sigma_{\xi \in \Omega} \mathcal{B}_\xi + \mathcal{B}^*_{\xi^*}) = \mathcal{B}_O$ (say).

□

Corollary 2:

Consider a GTS $S = (\Delta, \rho)$ where each operation has the properties stated in the previous result. Then $T_{ss'} \in \mathbf{EBS}_\Sigma(\sigma)$ where $\Sigma = \sigma_{ss'}$ and $\sigma = \sigma_s \cup \sigma_{s'}$.

Proof:

$$T_{ss'} = \bigvee_{O=(\Omega,\mathbf{p}) \in \Delta} \exists \mathbf{p} T^O_{ss'}(\mathbf{p})$$

Since $M \models T_{ss'}$, $M \models T^O_{ss'}(\mathbf{p})$ for some $O \in \Delta$ and some $\mathbf{p}_1 \in A^{|\mathbf{p}|}$. By **Claim 1**, $M|_{\sigma^O_{ss'}} \models T^O_{ss'}(\mathbf{p}_1)$. Then by **Result 2**, there exists $A_1 \subseteq A$ such that

1. $|A_1| \leq \mathcal{B}_O$
2. $\mathbf{p}_1 \in A_1^{|\mathbf{p}|}$
3. $\forall A_2, A_1 \subseteq A_2 \subseteq A$, there exists a $\sigma^O_{ss'}$ structure M'_2 with universe A_2 such that
   (a) $M'_2|_{\sigma_s \cup \sigma_{s'}} \subseteq M|_{\sigma^O_{ss'}}|_{\sigma_s \cup \sigma_{s'}}$
   (b) $M'_2 \models T^O_{ss'}(\mathbf{p}_1)$.

Consider any $\sigma_{ss'}$-structure M_2 such that $M_2|_{\sigma^O_{ss'}} = M_2'$. Then M_2 has the same universe A_2 as M_2' and
$M_2|_{\sigma_s \cup \sigma_{s'}} = M_2'|_{\sigma_s \cup \sigma_{s'}} \subseteq M|_{\sigma^O_{ss'}}|_{\sigma_s \cup \sigma_{s'}} = M|_{\sigma_s \cup \sigma_{s'}}$. Since $M_2|_{\sigma^O_{ss'}} = M_2'$, $M_2|_{\sigma^O_{ss'}} \models T^O_{ss'}(\mathbf{p}_1)$. Using **Claim 1**, then $M_2 \models T^O_{ss'}(\mathbf{p}_1)$ and hence $M_2 \models T_{ss'}$.

$|A_1| \leq \mathcal{B}_O \leq \mathbf{max}_{O \in \Delta} \mathcal{B}_O \;\; = \mathcal{B}$.

$\square$

Chapter 10

Deciding The Formulae of `InductionSolve`

10.1 Special GTSes, Safety Properties and Initial State Conditions

As seen in chapter 8, bounded model property for the formulae $IS1(k)$ and $IS2(k)$ allows us to *decide* their satisfiability. In this section, we show how for particular kinds of graph transforming systems, safety properties and initial state conditions having the bounded model property, we can use their individual bounded model properties to get bounded models for the formulae $IS1(k)$ and $IS2(k)$.

We firstly recall, that a GTS $S = (\Delta, \kappa)$ defines a transition relation on states $\mathcal{T}(s, s')$ as follows:

Given states s and s', $\mathcal{T}(s, s')$ is true iff there exists a $\sigma_{ss'}-$structure M such that $M|_{\sigma_s}$ defines s, $M|_{\sigma_{s'}}$ defines s' and $M \models T_{ss'}$.

We again recall that the safety property $\mathcal{P}$ and the initial-state conditions $\mathcal{I}$ are FO-expressible properties of states. Thus for a state s

- $\mathcal{P}(s)$ is true iff $s \models \varphi_P$ where φ_P is a σ_S-sentence expressing the safety property.
- $\mathcal{I}(s)$ is true iff $s \models \varphi_I$ where φ_I is a σ_S-sentence expressing the initial-state conditions.

We now consider the following special kind of graph-transformation systems, safety properties and initial-state conditions.
Consider the special kind of GTSes $S = (\Delta, \kappa)$ for which $T_{ss'} \in \mathbf{EBS}_{\Sigma}(\sigma)$ where $\Sigma = \sigma_{ss'}$

and $\sigma = \sigma_s \cup \sigma_{s'}$. Let $\mathcal{B}_\mathcal{T}$ denote the cardinal satisfying the conditions of the **EBS** property of $T_{ss'}$.
In the previous chapter, we already saw two kinds of GTSes which have the above property.

Let $\mathcal{Z} = \{\varphi \,|\varphi \in \mathbf{EBS}_{\Sigma_S}(\Sigma_S)\}$.
Now every formula φ can be uniquely encoded as a natural number $\langle\varphi\rangle$. Each of the symbols $\exists$, $\forall$, the boolean connectives, the parantheses (,), the equality symbol (=), the predicates of σ_S can be encoded by assigning unique numbers to them. The separation between adjacent symbols in the formula can be encoded as a unique number representing the separation. The variables can be encoded easily: If there are n variables $x_1, x_2, \ldots, x_n$ appearing in the formula, then we assign a unique number, say η for x_1 and then the x_i is encoded as the number $\eta\eta \ldots i$times. η can always be chosen so that this method of encoding variables doesnt produce numbers which already represent encodings of other symbols.
For example assign the following codes:

1. Quantifiers: $\exists = 11, \forall = 12$
2. Paranthesis: $(= 21,) = 22$
3. Boolean connectives: $\wedge = 31, \vee = 32, \neg = 33, \rightarrow = 34, \leftrightarrow = 35$
4. Equality(=): $= 4$
5. Predicates: $N = 5, E = 6$, $T_N^j = 7j (1 \leq j \leq \mathcal{N}_n)$, $T_E^j = 8j (1 \leq j \leq \mathcal{N}_e)$
6. The separator(space) $= 0$.
7. Variables: If there are n variables $x_1, x_2, \ldots, x_n$, then x_1 is encoded as 9, x_2 as 99, x_3 as 999 and so on

Then $\varphi = \exists x_1 \forall x_2 (N(x_1) \wedge E(x_1\ x_2))$ can be encoded as follows:
$\langle\varphi\rangle = 1109012099021050210902203106021090990220220$.
(Note that using the codes above, we can reconstruct a formula exactly from its encoding.)

Let, for a set $\mathcal{F}$ of formulae, $\langle\mathcal{F}\rangle = \{\langle\varphi\rangle \,|\varphi \in \mathcal{F}\}$. Then construct the function $f_S : \langle\mathcal{Z}\rangle \longrightarrow \aleph$ defined as

$$f_S(\langle\varphi\rangle) = \mathcal{B}_\varphi$$

where B_φ is the cardinal satisfying the conditions of the **EBS** property of φ.

Let $\mathcal{Z}_P$ be a subset of $\mathcal{Z}$ such that

1. It is closed under negation (for every $\varphi \in \mathcal{Z}_P$, $\neg\varphi \in \mathcal{Z}_P$)
2. $f_P = f_S|_{\langle\mathcal{Z}_P\rangle}$ (f_S restricted to $\langle\mathcal{Z}_P\rangle$) is a recursive function.

(Such a subset exists - Consider $\mathcal{Z}_P$ which contains only 2 sentences of $\mathcal{Z}$ which are negations of each other.)
Similarly let $\mathcal{Z}_I$ be a subset of $\mathcal{Z}$ such that

1. It is closed under negation (for every $\varphi \in \mathcal{Z}_I$, $\neg\varphi \in \mathcal{Z}_I$)
2. $f_I = f_S|_{\langle \mathcal{Z}_I \rangle}$ is a recursive function.

We then consider

- Safety properties $\mathcal{P}$ such that $\varphi_P \in \mathcal{Z}_P$
- Initial state conditions $\mathcal{I}$ such that $\varphi_I \in \mathcal{Z}_I$

An example of $\mathcal{Z}_P$ is the following:

$\mathcal{Z}_P^1$ = The set of all sentences which are of the form $\exists^*$ or $\forall^*$ in prenex normal form. Since the negation of an $\exists^*$ sentence is a $\forall^*$ sentence and vice-versa, $\mathcal{Z}_P^1$ is closed under negation. Further from **Claim 3**, we can see that any $\exists^*$ sentence and any $\forall^*$ sentence belongs to $\mathcal{Z}$. Hence $\mathcal{Z}_P^1 \subseteq \mathcal{Z}$. Further from **Claim 3**, for an $\exists^*$ sentence φ having m existential quantifiers, $\mathcal{B}_\varphi = m$ and for a $\forall^*$ sentence φ, $\mathcal{B}_\varphi = 0$. Then we can construct a procedure for computing $f_P(n)$.

The procedure takes as input n and first checks if n forms a valid encoding. If not, it returns **null**. If yes, then $n = \langle\varphi\rangle$ for some φ. It reconstructs the sentence φ from $\langle\varphi\rangle$. It brings the sentence in prenex normal form. It then checks whether the sentence is of the form $\exists^*$ or $\forall^*$. If its neither of these, it returns **null**. Else if the sentence is of the form $\exists^*$, then it counts the number(m) of existential quantifiers and returns m. Else if the sentence is of the form $\forall^*$, it returns 0.

10.2 Bounded Models For $IS1(k)$ and $IS2(k)$

We recall that

$$\begin{aligned} IS1(0) &= \mathcal{I}(s_1) \wedge \neg\mathcal{P}(s_1) \\ IS2(0) &= \neg\mathcal{P}(s_1) \end{aligned}$$

Consider $IS1(0)$.
Suppose there is a state s_1 with universe A such that $IS1(0)$ is true. Then

$$s_1 \models \varphi_I \wedge \neg\varphi_P$$

Then, from our assumptions about φ_I, there exists a subset $A_1 \subseteq A$ such that $|A_1| \leq f_I(\langle\varphi_I\rangle)$ satisfying the properties mentioned. Similarly for $\neg\varphi_P$, there exists a subset $A_2 \subseteq A$ such that $|A_2| \leq f_P(\langle\neg\varphi_P\rangle)$ and satisfying the properties mentioned. Construct

$$A' = A_1 \cup A_2$$

$|A'| \leq (|A_1| + |A_2|) \leq (f_I(\langle\neg\varphi_I\rangle) + f_P(\langle\neg\varphi_P\rangle))$.
Consider the substructure s_1' of s generated by A'. Then from assumptions about φ_I and φ_P,

$$s_1' \models \varphi_I \wedge \neg\varphi_P$$

Thus given s_1 satisfying $IS1(0)$, there is a state s_1' of bounded size ($|A'| \leq (f_I(\langle\neg\varphi_I\rangle) + f_P(\langle\neg\varphi_P\rangle)))$ satisfying $IS1(0)$.
We can similarly show that, given s_1 satisfying $IS2(0)$, there is a state s_1' of bounded size ($|A'| \leq f_P(\langle\neg\varphi_P\rangle))$ satisfying $IS2(0)$.

Let us consider $IS1(k)$ and $IS2(k)$ for $k \geq 1$. We have,

$$IS1(k) \;=\; \bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{i=2}^{i=k+1} \neg\mathcal{I}(s_i) \wedge \bigwedge_{1\leq i,j\leq k; i\neq j} (s_i \neq s_j) \wedge \mathcal{I}(s_1) \wedge \neg\mathcal{P}(s_{k+1})$$

$$IS2(k) \;=\; \bigwedge_{i=1}^{i=k} \mathcal{T}(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{i=k} \mathcal{P}(s_i) \wedge \bigwedge_{1\leq i,j\leq k; i\neq j} (s_i \neq s_j) \wedge \neg\mathcal{P}(s_{k+1})$$

We show now that if there exists a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$ (resp. $IS2(k)$), there exists a sequence of bounded states $s_1', s_2', \ldots, s_{k+1}'$ satisfying $IS1(k)$ (resp. $IS2(k)$). We note that since in `InductionSolve`, we proceed inductively i.e. checking $IS1(k)$ and $IS2(k)$ starting with $k = 0$ and increasing its value, while considering $k \geq 1$, we have already established that $\mathcal{I}(s) \rightarrow \mathcal{P}(s)$ (and hence $\neg\mathcal{P}(s) \rightarrow \neg\mathcal{I}(s)$) for any state s.

Consider a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$.
Since $\neg\mathcal{P}(s_{k+1})$ is true, $s_{k+1} \models \neg\varphi_P$. Let the universe of s_{k+1} be A. By assumption about $\neg\varphi_P$, there exists a subset $A_{k+1}^1 \subseteq A$ such that $|A_{k+1}^1| \leq f_P(\langle\neg\varphi_P\rangle)$ satisfying the properties mentioned. Let $A_{k+1} = A_{k+1}^1$.

Now consider s_k and s_{k+1}. Since $\mathcal{T}(s_k, s_{k+1})$ is true, there exists a $\sigma_{ss'}$−structure M_k such that $M_k|_{\sigma_s}$ defines s_k, $M_k|_{\sigma_s'}$ defines s_{k+1} and $M_k \models T_{ss'}$. (Thus M_k, s_k, s_{k+1} have the same universe, namely A.) Then by assumption, since $T_{ss'} \in \mathbf{EBS}_\Sigma(\sigma)(\Sigma = \sigma_{ss'}, \sigma = \sigma_s \cup \sigma_{s'})$, there exists $A_k^1 \subseteq A$ such that $|A_k^1| \leq \mathcal{B}_T$ and which satisfies the conditions of the **EBS** property.
Since $\mathcal{P}(s_k)$ is true, $s_k \models \varphi_P$ and hence by assumption, there is a subset $A_k^2 \subseteq A$ such that $|A_k^2| \leq f_P(\langle\varphi_P\rangle)$ and which satisfies the properties of $\mathcal{Z}_P$.

Further since $\neg\mathcal{I}(s_k)$ is true, $s_k \models \neg\varphi_I$ and hence by assumption, there is a subset $A_k^3 \subseteq A$ such that $|A_k^3| \leq f_I(\langle\neg\varphi_I\rangle)$ and which satisfies the properties of $\mathcal{Z}_I$.
Construct

$$A_k = A_k^1 \cup A_k^2 \cup A_k^3 \cup A_{k+1}$$

Note that

1. $A_{k+1} \subseteq A_k \subseteq A$

2. $|A_k| \leq |A_k^1| + |A_k^2| + |A_k^3| + |A_{k+1}|$
   $\leq \mathcal{B}_T + f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle) + |A_{k+1}|$

Now consider s_{k-1} and s_k. Since $\mathcal{T}(s_{k-1}, s_k)$ is true, there exists a $\sigma_{ss'}$−structure M_{k-1} such that $M_{k-1}|_{\sigma_s}$ defines s_{k-1}, $M_{k-1}|_{\sigma'_s}$ defines s_k and $M_{k-1} \models T_{ss'}$. (Thus M_{k-1}, s_{k-1}, s_k have the same universe, namely A.) Then by assumptions, since $T_{ss'} \in \mathbf{EBS}_\Sigma(\sigma)(\Sigma = \sigma_{ss'}, \sigma = \sigma_s \cup \sigma_{s'}$, there exists $A_{k-1}^1 \subseteq A$ such that $|A_{k-1}^1| \leq \mathcal{B}_T$ satisfying the conditions of the **EBS** property.
Since $\mathcal{P}(s_{k-1})$ is true, $s_{k-1} \models \varphi_P$ and hence by assumption, there is a subset $A_{k-1}^2 \subseteq A$ such that $|A_{k-1}^2| \leq f_P(\langle\varphi_P\rangle)$ and which satisfies the properties of $\mathcal{Z}_P$.
Further since $\neg\mathcal{I}(s_{k-1})$ is true, $s_{k-1} \models \neg\varphi_I$ and hence by assumption, there is a subset $A_{k-1}^3 \subseteq A$ such that $|A_{k-1}^3| \leq f_I(\langle\neg\varphi_I\rangle)$ and which satisfies the properties of $\mathcal{Z}_I$.
Construct

$$A_{k-1} = A_{k-1}^1 \cup A_{k-1}^2 \cup A_{k-1}^3 \cup A_k$$

Note that

1. $A_k \subseteq A_{k-1} \subseteq A$

2. $|A_{k-1}| \leq |A_{k-1}^1| + |A_{k-1}^2| + |A_{k-1}^3| + |A_k|$
   $\leq \mathcal{B}_T + f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle) + |A_k|$

We continue this way back to s_1. At s_1, we construct A_1^1 as we constructed any A_i^1. Now since $\mathcal{I}(s_1)$ is true, $s_1 \models \varphi_I$ and hence by assumption, there is a subset $A_1^2 \subseteq A$ such that $|A_1^2| \leq f_I(\langle\varphi_I\rangle)$ and which satisfies the properties of $\mathcal{Z}_I$.
Then construct

$$A_1 = A_1^1 \cup A_1^2 \cup A_2$$

Note that

1. $A_2 \subseteq A_1 \subseteq A$

2. $|A_1| \leq |A_1^1| + |A_1^2| + |A_2|$
   $\leq \mathcal{B}_T + f_I(\langle\varphi_I\rangle) + |A_2|$

(Note that since, we have already established that $\mathcal{I}(s) \to \mathcal{P}(s)$, we do not consider the $f_P(\langle\varphi_P\rangle)$ term for $|A_1|$. By the same reason, we do not consider the$f_I(\langle\neg\varphi_I\rangle)$ term for

$|A_{k+1}|$)

Thus we have a sequence of A_is satisfying the following recurrence relations

1. $A_i = A_i^1 \cup A_i^2 \cup A_i^3 \cup A_{i+1} \quad 1 \leq i \leq k$
   $A_1 = A_1^1 \cup A_1^2 \cup A_2$
2. $A_{i+1} \subseteq A_i \subseteq A \qquad 1 \leq i \leq k$
3. $$\begin{array}{lll} |A_i| & \leq & \mathcal{B}_T + f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle) + |A_{i+1}| \qquad 2 \leq i \leq k \\ |A_{k+1}| & \leq & f_P(\langle\neg\varphi_P\rangle) \\ |A_1| & \leq & \mathcal{B}_T + f_I(\langle\varphi_I\rangle) \end{array}$$

Consider s_i and $s_j (1 \leq i, j \leq k+1, i \neq j)$. s_i and s_j have the same universe, namely A. Now since $s_i \neq s_j$, there exists $A_{ij} \subseteq A$ such that the substructure s_i^1 of s_i generated by A_{ij} is not equal to the substructure s_j^1 of s_j generated by A_{ij}, i.e. $s_i^1 \neq s_j^1$.
Consider A_{ij}^* such that

$$|A_{ij}^*| = min\left\{|A_{ij}| \; |A_{ij} \subseteq A, s_i^1 \neq s_j^1\right\}$$

Since σ_S consists of only unary and binary predicates, we can see that

$$|A_{ij}^*| \leq 2 \qquad 1 \leq i, j \leq k+1, i \neq j$$

Construct

$$A' = A_1 \cup \bigcup_{1 \leq i,j \leq k+1, i \neq j} A_{ij}^*$$

<u>Size Calculation for A'</u>
Solving the recurrence relation

$$\begin{array}{l} |A_i| \leq \mathcal{B}_T + f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle) + |A_{i+1}| \quad 2 \leq i \leq k \\ |A_{k+1}| = f_P(\langle\neg\varphi_P\rangle) \end{array}$$

we get

$$|A_i| \leq (k+1-i)(\mathcal{B}_T + f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle)) + f_P(\langle\neg\varphi_P\rangle)$$

Then

$$\begin{array}{lll} |A'| & \leq & |A_1| + \Sigma_{1 \leq i,j \leq k+1, i \neq j} |A_{ij}^*| \\ & \leq & \mathcal{B}_T + f_I(\langle\varphi_I\rangle) + |A_2| + \Sigma_{1 \leq i,j \leq k+1, i \neq j} 2 \\ & \leq & k.\mathcal{B}_T + (k-1).(f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle)) + \\ & & f_I(\langle\varphi_I\rangle) + f_P(\langle\neg\varphi_P\rangle) + 2.{}^{k+1}C_2 \end{array}$$

Thus A' is of bounded size.

Now for each $i, 1 \leq i \leq k+1$, consider M_i (seen earlier) such that $M_i \models T_{ss'}$ and $M_i|_{\sigma_s}$ defines s_i and $M_i|_{\sigma'_s}$ defines s_{i+1}. M_i has universe A. Consider A_i^1. Since $A_i^1 \subseteq A' \subseteq A$, from the assumptions about the GTS, there exists a $\sigma_{ss'}-$structure M_i^1 such that

1. $M_i^1|_{\sigma_s \cup \sigma_{s'}} \subseteq M_i|_{\sigma_s \cup \sigma_{s'}}$

2. $M_i^1 \models T_{ss'}$

Consider s'_i and s'_{i+1} such that $M_i^1|_{\sigma_s}$ defines s'_i and $M_i^1|_{\sigma_{s'}}$ defines s'_{i+1}.

1. Since $M_i^1 \models T_{ss'}$, $\mathcal{T}(s'_i, s'_{i+1})$ is true.
2. Since $M_i^1|_{\sigma_s \cup \sigma_{s'}} \subseteq M_i|_{\sigma_s \cup \sigma_{s'}}$, $M_i^1|_{\sigma_s} \subseteq M_i|_{\sigma_s}$. Thus $s'_i \subseteq s_i$. Similarly $s'_{i+1} \subseteq s_{i+1}$.
3. For $2 \leq i \leq k, \mathcal{P}(s_i)$ is true, hence $s_i \models \varphi_P$. Further $A_i^2 \subseteq A_i \subseteq A' \subseteq A$ and s'_i (having universe A') is a substructure of s_i (having universe A). Then from assumptions about φ_P, $s'_i \models \varphi_P$. Hence $\mathcal{P}(s'_i)$ is true.
4. For $2 \leq i \leq k, \neg\mathcal{I}(s_i)$ is true, hence $s_i \models \neg\varphi_I$. Further $A_i^3 \subseteq A_i \subseteq A' \subseteq A$ and s'_i (having universe A') is a substructure of s_i (having universe A). Then from assumptions about $\neg\varphi_I$, $s'_i \models \neg\varphi_I$. Hence $\neg\mathcal{I}(s'_i)$ is true.
5. Consider $i = k+1$. Since $\neg\mathcal{P}(s_{k+1})$ is true, $s_{k+1} \models \neg\varphi_P$. Further $A_{k+1} \subseteq A_1 \subseteq A' \subseteq A$ and s'_{k+1} (having universe A') is a substructure of s_i (having universe A). Then from assumptions about $\neg\varphi_P$, $s'_{k+1} \models \neg\varphi_P$. Hence $\neg\mathcal{P}(s'_{k+1})$ is true.
6. Consider $i = 1$. Since $\mathcal{I}(s_1)$ is true, $s_1 \models \varphi_I$. Further $A_1^2 \subseteq A_1 \subseteq A' \subseteq A$ and s'_1 (having universe A') is a substructure of s_1 (having universe A). Then from assumptions about φ_I, $s'_1 \models \varphi_I$. Hence $\mathcal{I}(s'_1)$ is true.

Since we already established that $\mathcal{I}(s) \rightarrow \mathcal{P}(s)$, we have

1. $\mathcal{I}(s'_1) \rightarrow \mathcal{P}(s'_1)$ and hence $\mathcal{P}(s'_1)$ is true.
2. $\neg\mathcal{P}(s'_{k+1}) \rightarrow \neg\mathcal{I}(s'_{k+1})$ and hence $\neg\mathcal{I}(s'_{k+1})$ is true.

Finally for $1 \leq i, j \leq k+1, i \neq j$, A_{ij}^* is such that the substructure s_i^* of s_i generated by A_{ij}^* and the substructure s_j^* of s_j generated by A_{ij}^* are such that $s_i^* \neq s_j^*$. Now since $A_{ij}^* \subseteq A' \subseteq A$, $s_i^* \subseteq s'_i$ and $s_j^* \subseteq s'_j$. Since $s_i^* \neq s_j^*$, we have $s'_i \neq s'_j$.

Thus we have shown that if there is a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$, there exists a sequence of states $s'_1, s'_2, \ldots, s'_{k+1}$ satisfying $IS1(k)$ such that each s'_i is bounded. Each s'_i has universe A' which is bounded by $\mathcal{B}$ given by

$$\mathcal{B} = k.\mathcal{B}_T + (k-1).(f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle)) + f_I(\langle\varphi_I\rangle) + f_P(\langle\neg\varphi_P\rangle) + 2.^{k+1}C_2$$

It can be shown that the $2.^{k+1}C_2$ term can be replaced by a $2.(k-1)$ term so that the bound is given by

$$\mathcal{B} = k.\mathcal{B}_T + (k-1).(f_P(\langle\varphi_P\rangle) + f_I(\langle\neg\varphi_I\rangle) + 2) + f_I(\langle\varphi_I\rangle) + f_P(\langle\neg\varphi_P\rangle)$$

In a very similar fashion, we can show that if there is a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS2(k)$, there exists a sequence of states $s'_1, s'_2, \ldots, s'_{k+1}$ satisfying $IS2(k)$ such that each s'_i is bounded. Each s'_i has universe A' which is bounded by $\mathcal{B}$ given by

$$\mathcal{B} \leq k.(\mathcal{B}_T + f_P(\langle\varphi_P\rangle)) + f_P(\langle\neg\varphi_P\rangle)$$

Thus for $IS1(k)$ (resp. $IS2(k)$), if we can find $s'_1, s'_2, \ldots, s'_{k+1}$ having universes of size bounded (by the bound we calculated above) satisfying $IS1(k)$ (resp. $IS2(k)$), then we have found a sequence of states satisfying $IS1(k)$ (resp. $IS2(k)$). If however, we cannot find $s'_1, s'_2, \ldots, s'_{k+1}$ having universes of size bounded (by the bound we calculated above) satisfying $IS1(k)$ (resp. $IS2(k)$), then there cannot exist a sequence of states $s_1, s_2, \ldots, s_{k+1}$ satisfying $IS1(k)$ (resp. $IS2(k)$), since if so, we should have been able to find $s'_1, s'_2, \ldots, s'_{k+1}$ having universes of bounded size satisfying $IS1(k)$ (resp. $IS2(k)$).

10.3 Decision Procedure For `InductionSolve`

Using the results of the previous section, we can now construct a decision procedure for checking the validity of the formulae appearing in `InductionSolve`. We denote the formulae of `InductionSolve` by $ISF1(k)$ and $ISF2(k)$ where $ISF1(k) = \neg IS1(k)$ and $ISF2(k) = \neg IS2(k)$.

The procedure takes in as input the sentences $T_{ss'}, \varphi_P$ and φ_I, k (value of iterator in `InductionSolve`) and J ($J = 1, 2$ which indicates which formula to decide i.e. $ISF1$ or $ISF2$) as input. It is assumed that $\mathcal{B}_T$ and the functions f_I and f_P are known. The procedure then performs the following operations:

1. Formula Generation
   (a) For each $i, 1 \leq i \leq k$, it produces the sentence $T_{s_i,s_{i+1}}$ as follows:
      i. It replaces each occurence of the predicates N_s and $N_{s'}$ with N_{s_i} and $N_{s_{i+1}}$. It does likewise for predicates E_s, $E_{s'}$, $T^j_{N,s}$, $T^j_{N,s'}(1 \leq j \leq \mathcal{N}_n)$, $T^j_{E,s}$, $T^j_{E,s'}(1 \leq j \leq \mathcal{N}_e)$.

ii. For each operation $O = (\Omega, \mathbf{p}) \in \Delta$ (where the GTS is $S = (\Delta, \kappa)$), for each $\xi \in \Omega$, it replaces each occcurence of

A. X_n^{ξ} with X_{n,s_i}^{ξ} , $\lambda_n \in \kappa_N$

B. Y_e^{ξ} with Y_{e,s_i}^{ξ} , $\mu_e \in \kappa_E$

C. X_n^{O} with X_{n,s_i}^{O} , $\lambda_n \in \kappa_N$

D. Y_e^{O} with Y_{e,s_i}^{O} , $\mu_e \in \kappa_E$

(b) For each $i, 1 \leq i \leq k{+}1$, it produces the sentence φ_{P,s_i} from φ_P by replacing each occurence of the predicate N with N_{s_i}, E with E_{s_i}, T_N^j with $T_{N,s_i}^j (1 \leq j \leq \mathcal{N}_n)$, and T_E^j with $T_{E,s_i}^j (1 \leq j \leq \mathcal{N}_e)$.

(c) For each $i, 1 \leq i \leq k+1$, it produces the sentence φ_{I,s_i} from φ_I as in (2).

(d) For $1 \leq i, j \leq k+1, i \neq j$, it produces the sentence

$$Neq_{s_i,s_j} = \neg \left(\begin{array}{lll} \forall x & (N_{s_i}(x) \leftrightarrow N_{s_j}(x) \wedge & \bigwedge_{k=1}^{k=\mathcal{N}_n} T_{N,s_i}^k(x) \leftrightarrow T_{N,s_j}^k(x) \ ) \\ \forall x, y & (E_{s_i}(x,y) \leftrightarrow E_{s_j}(x,y) \wedge & \bigwedge_{k=1}^{k=\mathcal{N}_e} T_{E,s_i}^k(x,y) \leftrightarrow T_{E,s_j}^k(x,y) \ ) \end{array} \right)$$

(e)
- If $J = 1$, it produces the sentence
  $IS1_k =$

$$\bigwedge_{i=1}^{i=k} \mathcal{T}_{s_i,s_{i+1}} \wedge \bigwedge_{i=1}^{i=k} \varphi_{P,s_i} \wedge \bigwedge_{i=2}^{i=k+1} \neg\varphi_{I,s_i} \wedge \bigwedge_{1 \leq i,j \leq k+1; i \neq j} Neq_{s_i,s_j} \wedge \varphi_{I,s_1} \wedge \neg\varphi_{P,s_{k+1}}$$

- Else if $J = 2$, it produces the sentence

$$IS2_k = \bigwedge_{i=1}^{i=k} \mathcal{T}_{s_i,s_{i+1}} \wedge \bigwedge_{i=1}^{i=k} \varphi_{P,s_i} \wedge \bigwedge_{1 \leq i,j \leq k+1; i \neq j} Neq_{s_i,s_j} \wedge \neg\varphi_{P,s_{k+1}}$$

2. Removal of equality.

It then removes equality (=) between variables as follows:

Let σ be the set of all predicates appearing in ISJ_k. Let P be a $k-$ary predicate in σ. Then $x = y$ is replaced with

$$\bigwedge_{P \in \sigma} \forall z_1, z_2, \ldots, z_{k-1} \quad \left[\begin{array}{ccc} P(x, z_1, z_2, \ldots, z_{k-1}) & \leftrightarrow & P(y, z_1, z_2, \ldots, z_{k-1}) \wedge \\ P(z_1, x, z_2, \ldots, z_{k-1}) & \leftrightarrow & P(z_1, y, z_2, \ldots, z_{k-1}) \wedge \\ & \vdots & \\ P(z_1, z_2, \ldots, z_{k-1}, x) & \leftrightarrow & P(z_1, z_2, \ldots, z_{k-1}, y) \end{array} \right]$$

Let ISJ_k^1 be the resulting formula after removal of equalities.

3. Removal of negations preceding quantifiers.

   It removes negations just preceding quantifiers using the rules $\neg\exists = \forall\neg$ and $\neg\forall = \exists\neg$ to produce ISJ_k^2.

4. Bound computation.

   It then computes the bound $\mathcal{B}$ for $ISJ(k)$ using the expression derived earlier. Since $\mathcal{B}_T$ is known and since f_P, *and* f_I *are recursive functions, it computes* $\mathcal{B}$ *by using the procedures for these.* It then constructs the set $A = \{a_1, a_2, \ldots, a_{\mathcal{B}}\}$.

5. Elimination of quantifiers.

   It eliminates quantifiers.

   - Each occurence of $\exists x$ is replaced with $\bigvee_{x \in A}$.
   - Each occurence of $\forall x$ is replaced with $\bigwedge_{x \in A}$.

   Let the resulting formula be ISJ_k^3.
   Note that after this step there are no quantifiers or variables in ISJ_k^3.

6. Propositionalization.

   It propositionalises ISJ_k^3 to produce ISJ_k^4.
   For each $P \in \sigma$, $P, k-$ary, it replaces $P(a_{i_1}, a_{i_2}, \ldots, a_{i_k})(1 \leq i_j \leq \mathcal{B}, 1 \leq j \leq k)$ with the propositional variable $P_{a_{i_1}, a_{i_2}, \ldots, a_{i_k}}$.

7. Invoking a propositional SAT solver.

   The formula ISJ_k^4 is a propositional logic formula and is fed to a propositional SAT solver which *decides* the satisfiability of the formula.

8. Output.

   If the SAT solver returns **True**, it means ISJ_k^4 and hence ISJ_k is satisfiable. Then $ISFJ(k)$ is not valid and hence the procedure outputs **False**. The satisfying assignment produced by the SAT solver gives the counterexample for $ISFJ(k)$.
   If the SAT solver returns **False**, it means ISJ_k^4 and hence ISJ_k is unsatisfiable. Then $ISFJ(k)$ is valid and hence the procedure outputs **True**.

The entire flow is shown in the figure below.

10.4 Checking the Library System

We have already shown in chapter 9 that the library system can be expressed as a GTS which is of the kind mentioned in the previous chapter. We now look at the safety property and the initial state conditions.
The library system starts from the empty graph. Thus $\mathcal{I}$ is expressed by the formula φ_I which is given as

$$\begin{aligned}\varphi_I &= \forall x \neg N(x) \wedge \forall x, y, \neg E(x,y)\\ &= \forall x, y(\neg N(x) \wedge \neg E(x,y))\end{aligned}$$

Thus φ_I is a $\forall^*$ formula. Thus φ_I belongs to the example $\mathcal{Z}_P^1$ which we had considered in 10.1.

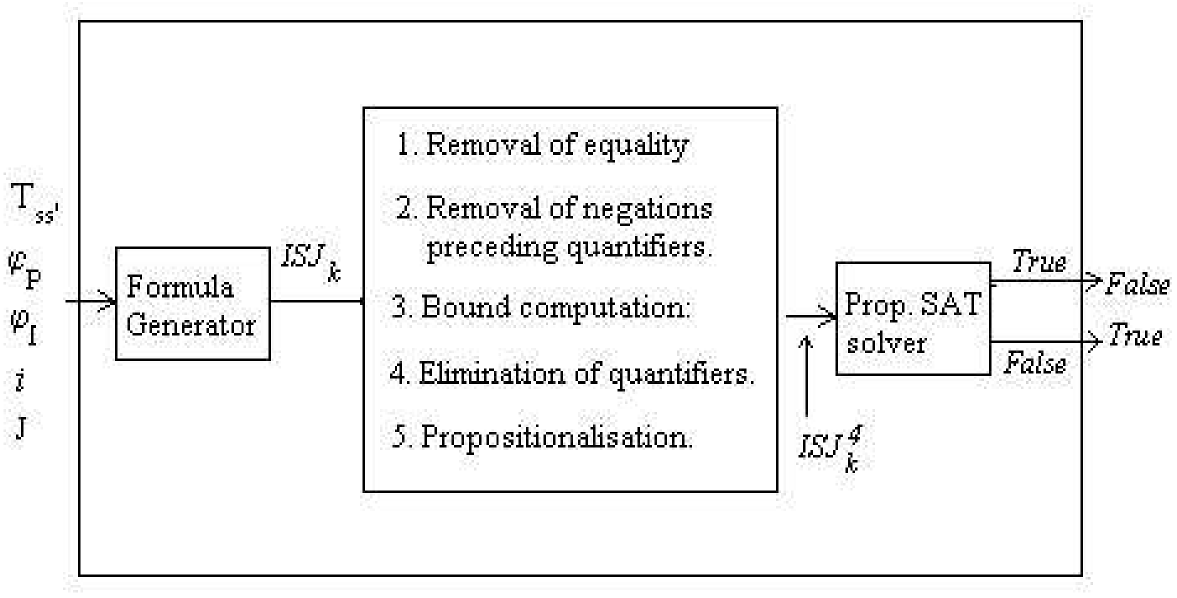
The safety property $\mathcal{P}$ we wanted to check was the $\mathcal{C}$-$\mathcal{T}$-$\mathcal{B}$-$\mathcal{S}$ pattern. This pattern is present in a state if there is a Claim instance connected to a Title instance which is connected to a Book instance which in turn is connected to a Shelf instance. If this pattern is present it means that a claim for a title is kept pending even when there is a book for the title on shelf which is clearly an undesirable situation. We can express this as a safety property by the formula φ_P which is given as
$\varphi_P =$

$$\neg \exists c,t,b,s\, (\mathcal{C}(c) \wedge \mathcal{T}(t) \wedge \mathcal{B}(b) \wedge \mathcal{S}(s) \wedge N(c) \wedge N(t) \wedge N(b) \wedge N(s) \wedge E(c,t) \wedge E(b,t) \wedge E(b,s))$$
$$\forall c,t,b,s \neg\, (\mathcal{C}(c) \wedge \mathcal{T}(t) \wedge \mathcal{B}(b) \wedge \mathcal{S}(s) \wedge N(c) \wedge N(t) \wedge N(b) \wedge N(s) \wedge E(c,t) \wedge E(b,t) \wedge E(b,s))$$

Thus φ_P is also a $\forall^*$ formula. Thus φ_P also belongs to the example $\mathcal{Z}_P^1$ which we had considered in 10.1.

We just proved that for the kind of GTSes, safety properties and initial state conditions mentioned in 10.1, we can decide the formulae $IS1(k)$ and $IS2(k)$. The library system, the $\mathcal{C}$-$\mathcal{T}$-$\mathcal{B}$-$\mathcal{S}$ safety property and the initial graph of the library system - are all of the kind mentioned in 10.1. Thus we can decide the formulae $IS1(k)$ and $IS2(k)$ for the library system.

For the $\mathcal{C}$-$\mathcal{T}$-$\mathcal{B}$-$\mathcal{S}$ safety property, $f_P(\langle\varphi_P\rangle) = 0$ and $f_P(\langle\neg\varphi_P\rangle) = 4$. For the initial state, $f_I(\langle\varphi_I\rangle) = 0$ and $f_I(\langle\neg\varphi_I\rangle) = 2$. For the library system, $\mathcal{B}_T = 8$. The bounds for $k \geq 1$ for the system were - (a) $IS1(k) : B = 12k - 1$ (b) $IS2(k) : B = 8k + 3$. Though in general we cannot put bounds on the iterator k of `InductionSolve`, it turns out that for $k = 1$ itself, `InductionSolve` terminates for the library system returning **True**, meaning that the library system does satisfy the $\mathcal{C}$-$\mathcal{T}$-$\mathcal{B}$-$\mathcal{S}$ safety property.



Validity Checker

Chapter 11

Some studies on the EBS property

11.1 A Syntactic Subclass of $\mathcal{EBS}$

(**Note:** There is a modification to the definition of the **QPS** property and a slight modification to Example 1, and the statements of Lemmas 4 and 5 as given in the paper (submission no: 139) submitted to FSTTCS '07. These should be as given below.)

Without loss of generality, consider a formula $\varphi(\mathbf{x})$ in prenex normal form with all variables named uniquely. Let $V(\varphi(\mathbf{x}))$ be the set of free variables and variables that are existentially quantified before (to the left of) all universally quantified variables in the prefix structure of $\varphi(\mathbf{x})$. Let $AV(\varphi(\mathbf{x}))$ be the set of all $\forall$ quantified variables of $\varphi(\mathbf{x})$ and $EV(\varphi(\mathbf{x}))$ be the set of all $\exists$ quantified variables of $\varphi(\mathbf{x})$ not in $V(\varphi(\mathbf{x}))$. Further, we index the quantified variables in the prefix in increasing order from left to right.

Definition 11.1 (QPS property) *The formula $\varphi(\boldsymbol{x})$ is said to satisfy the **Quantifier based Predicate Separable** property with respect to $\sigma \subseteq \Sigma$ (denoted as $\varphi(\boldsymbol{x}) \in \boldsymbol{QPS}_\Sigma(\sigma)$) if $Sig(\varphi) = \Sigma$ and the following conditions hold:*

1. *Every predicate $P \in \Sigma$ of arity $k \geq 1$ satisfies the following:*
   - *If $P \in \sigma$, then every argument of every instance of P is in $V(\varphi(\boldsymbol{x})) \cup AV(\varphi(\boldsymbol{x}))$.*
   - *If $P \in \Sigma \setminus \sigma$, then for each i in 1 through k, either (a) the i^{th} argument of every instance of P in φ is in $V(\varphi(\boldsymbol{x})) \cup AV(\varphi(\boldsymbol{x}))$, or (b) the i^{th} argument of every instance of P in φ is in $V(\varphi(\boldsymbol{x})) \cup EV(\varphi(\boldsymbol{x}))$. Further, for each instance of P, either (a) all its arguments are from $V(\varphi(\boldsymbol{x})) \cup AV(\varphi(\boldsymbol{x}))$ or (b) its highest indexed argument is from $EV(\varphi(\boldsymbol{x}))$.*
2. *For any 2 instances of $P \in \Sigma \setminus \sigma$ of arity ≥ 1 having non-indentical argument lists, either (a) all arguments of each instance are from $V(\varphi(\boldsymbol{x})) \cup AV(\varphi(\boldsymbol{x}))$ or (b) there exists a variable from $EV(\varphi(\boldsymbol{x}))$ which appears as the i^{th} argument of one instance but not as the i^{th} argument of the other.*

Example 1:
In the formula $\varphi(u)$ as given in the paper, the $\neg Q(z)$ should actually be $\neg Q(x)$. The rest of the formula remains unchanged.

Lemma 4:

Let $\varphi(\mathbf{x})$ be a formula in prenex normal form having the $\exists^*\forall^*\exists^*$ quantifier prefix, in which every predicate of arity ≥ 1 in $\Sigma = Sig(\varphi)$ appears exactly once. Then there exists $\sigma \subseteq \Sigma$ such that $\varphi(\mathbf{x}) \in \mathbf{QPS}_\Sigma(\sigma)$.

Proof: Follows from definition.

Lemma 5:

Let $\varphi(\mathbf{x})$ be a formula in $QPS_\Sigma(\sigma)$ in prenex normal form. Let k be the total number of instances of all predicates of arity ≥ 1 in $\varphi(\mathbf{x})$ and k_0 be the total number of nullary predicate instances in $\varphi(\mathbf{x})$. Let $r = |EV(\varphi(\mathbf{x}))|$. Then $\varphi(\mathbf{x})$ satisfies all conditions in **EBS** Definition with $\mathcal{B} = k_0 + |\mathbf{x}| + r.3^k$.

Proof:

Consider a $\Sigma-$structure M such that $M \models \varphi(\mathbf{b}_1)$ and such that $|A| \geq \mathcal{B} = k_0+|\mathbf{x}|+r.3^k$. Let $\mathbf{b}_2$ be the witnesses to the leftmost $\exists$ quantified variables of φ. Consider any subset A_1 of A such that $\mathbf{b}_1 \in A_1^{|\mathbf{x}|}, \mathbf{b}_2 \in A_1^{|\mathbf{b}_2|}$ and such that in addition to these elements (i.e. different from these) A_1 has $r.3^k$ distinct elements of A. Then $|A_1| \leq \mathcal{B}$. Let the $r.3^k$ elements be denoted as $a_{(1,1)}, \ldots, a_{(1,3^k)}, a_{(2,1)}, \ldots, a_{(2,3^k)}, \ldots, a_{(r,1)}, \ldots, a_{(r,3^k)}$. Consider any A_2 such that $A_1 \subseteq A_2 \subseteq A$.

Let $\varphi(\mathbf{x})$ be of the form

$$\varphi(\mathbf{x}) = \exists\mathbf{y}\forall\mathbf{z}_0\exists v_1\forall\mathbf{z}_1\exists v_2\forall\mathbf{z}_2 \ldots \exists v_r\forall\mathbf{z}_r\psi(\mathbf{x}, \mathbf{y}, \mathbf{z}_0, v_1, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$$

where the matrix ψ is quantifier-free. We now show how to choose $v_1, \ldots, v_r$ and how to define the predicates of $\Sigma \setminus \sigma$ so that we can get the desired Σ-structure M_2 satisfying the **EBS** conditions. Choose $\mathbf{x} = \mathbf{b}_1$ and $\mathbf{y} = \mathbf{b}_2$. Recall that $A_2 \subseteq A$.

We move from left to right in the quantifier prefix of φ and observe the truth values in M of the predicate instances of ψ as we substitute values from A_2 for the $\forall$ quantified variables and values from A for the $\exists$ quantified variables. Firstly we substitute the values for the variables of $V(\varphi(\mathbf{x}))$, namely $\mathbf{b}_1$ for $\mathbf{x}$ and $\mathbf{b}_2$ for $\mathbf{y}$. Next we substitute values for $\mathbf{z}_0$ from A_2. For each value $\mathbf{d}_0 \in A_2^{|\mathbf{z}_0|}$ of $\mathbf{z}_0$, there is a certain value say e_1 of v_1 ($e_1 \in A$) which is used to make $\varphi(\mathbf{b}_1)$ true in M. Upon substitution of e_1 for v_1 in ψ, each predicate instance when interpreted in M with the (possibly incompletely specified) values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$, can be in one of 3 'states': (a) **True** (b) **False** (c) Undetermined at that stage (because the values of the subsequent variables in the prefix sequence of φ have yet not been specified.) Since each predicate instance is in one of these 3 states, the states of all predicate instances taken together determines a 'configuration' of states. Since there are k predicate instances, there are 3^k configurations. Number the 3^k configurations from 1 to 3^k. Then upon substitution of value e_1 for v_1, let

j_1 be the configuration of the states of all predicates instances, when interpreted in M, for the values of their arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$. Then from A_2, we choose the value $a_{(1,j_1)}$ to substitute for v_1. Further we define the interpretations in M_2 of the predicates of Σ such that the states of the predicate instances, when interpreted in M_2, for the values of their arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$, produce the same configuration, namely j_1. In particular, each predicate instance, when interpreted in M_2, with the (possibly incompletely specified) values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$, has the same state as it has when interpreted in M with the (possibly incompletely specified) values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{z}_1, \ldots, v_r, \mathbf{z}_r)$.

Next we substitute values for $\mathbf{z}_1$ from A_2. For each value $\mathbf{d}_1 \in A_2^{|\mathbf{z}_1|}$ of $\mathbf{z}_1$, there is a certain value say e_2 of v_2 ($e_2 \in A$) which is used to make $\varphi(\mathbf{b}_1)$ true in M. Upon substitution of value e_2 for v_2, let j_2 be the configuration of the states of all predicates instances, when interpreted in M, for the values of their arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{d}_1, e_2, \mathbf{z}_2, \ldots, v_r, \mathbf{z}_r)$. Then from A_2, we choose the value $a_{(2,j_2)}$ to substitute for v_2. Further we define the interpretations in M_2 of the predicates of Σ such that the states of the predicate instances, when interpreted in M_2, for the values of their arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \mathbf{z}_2, \ldots, v_r, \mathbf{z}_r)$, produce the same configuration, namely j_1. In particular, each predicate instance, when interpreted in M_2, with the (possibly incompletely specified) values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \mathbf{z}_2, \ldots, v_r, \mathbf{z}_r)$ has the same state as it has when interpreted in M with the (possibly incompletely specified) values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{d}_1, e_2, \mathbf{z}_2, \ldots, v_r, \mathbf{z}_r)$

We proceed this way till eventually for each $\mathbf{d}_0 \in A_2^{|\mathbf{z}_0|}, \ldots, \mathbf{d}_r \in A_2^{|\mathbf{z}_r|}$, we would have determined the values a_{i,j_i} for $v_i (1 \leq i \leq r)$ and would have defined the state of each predicate instance, when interpreted in M_2, for the values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$. In particular, since in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$ all predicate instances have values for all arguments specified, we would have defined the truth values in M_2 of each predicate instance, for the values of its arguments as they appear in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$. Note that if for a predicate instance, all arguments are from $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$, then its interpretation in M_2 is exactly the same as in M restricted to A_2.

A *conflict* is said to occur for a predicate P if for some set of values of its arguments, it has been defined to be both **True** and **False**. The question that now arises is: Does the interpretation of the predicates in M_2, the way we have constructed above, contain a conflict for any predicate?

Now a conflict can occur in two ways:

1. Internal: Given values $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_r$, there are 2 different instances I_1 and I_2 of the same predicate which have the same set of values for their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$ but (w.l.o.g) evaluate to **True** and **False**

respectively, when interpreted in M_2.

2. External: Given values $\mathbf{d}_{01}, \mathbf{d}_{11}, \ldots, \mathbf{d}_{r1}$ and $\mathbf{d}_{02}, \mathbf{d}_{12}, \ldots, \mathbf{d}_{r2}$ (and hence the corresponding values $a_{(11,j_{11})} \ldots a_{(r1,j_{r1})}$ and $a_{(12,j_{12})} \ldots a_{(r2,j_{r2})}$ to the variables $v_1, \ldots, v_r$), there are 2 instances I_1 and I_2 of the same predicate such that I_1 has the same set of values for its arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_{01}, a_{(11,j_{11})}, \mathbf{d}_{11}, a_{(21,j_{21})}, \ldots, a_{(r1,j_{r1})}, \mathbf{d}_{r1})$ as I_2 has for its arguments in
$\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_{02}, a_{(12,j_{12})}, \mathbf{d}_{12}, a_{(22,j_{22})}, \ldots, a_{(r2,j_{r2})}, \mathbf{d}_{r2})$ but (w.l.o.g) evaluate to **True** and **False** respectively when interpreted in M_2.

We now show that both of the above do not occur.

Consider 1. Suppose there are 2 different instances I_1 and I_2 of a predicate P which have the same set of values to their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$ but the former is **True** and latter **False** when interpreted in M_2. Since $\varphi \in QPS_\Sigma(\sigma)$, if $P \in \sigma$, then since all its arguments in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$, as seen earlier, it is interpreted in M_2 exactly as in M restricted to A_2. Then this conflict must have been present in M also, which is not the case. If $P \in \Sigma \setminus \sigma$, then since $\varphi \in \mathbf{QPS}_\Sigma(\sigma)$, we have the following cases:

1. All arguments of both I_1 and I_2 are from $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$. Then as seen earlier, their interpretations in M_2 are exactly as in M restricted to A_2. Then this conflict must have been present in M too, which is not the case.

2. I_1 and I_2 have different argument lists. Then there is a variable v_s from $EV(\varphi(\mathbf{x}))$ which appears as the i^{th} argument of I_1 but not as the i^{th} argument of I_2. Now the i^{th} argument of I_2 is a variable $v \in V(\varphi(\mathbf{x})) \cup EV(\varphi(\mathbf{x}))$. Then the value substituted for v_s is $a_{s,u}$ and for v is either some b value or some $a_{t,w}$ $(t \neq s)$. Either of the latter 2 values of v are different from $a_{s,u}$. Clearly I_1 and I_2 then do not have the same set of values to their arguments in which is a contradiction.

3. I_1 and I_2 have the same argument lists. Since I_1 and I_2, for the values to their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$ were **True** and **False** when interpreted in M_2, they were were **True** and **False** respectively when interpreted in M, for the values of their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{d}_1, e_2, \ldots, e_r, \mathbf{d}_r)$. (the interpretations in M_2 of I_1 and I_2 were defined this way.) But since I_1 and I_2 have the same argument lists, they had the same values to their variables in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{d}_1, e_2, \ldots, e_r, \mathbf{d}_r)$. This means there is a conflict in M.

Thus there are no internal conflicts.

Consider 2. Suppose there are 2 instances I_1 and I_2 of P such that I_1 in
$\phi_1 = \psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_{01}, a_{(11,j_{11})}, \mathbf{d}_{11}, a_{(21,j_{21})}, \ldots, a_{(r1,j_{r1})}, \mathbf{d}_{r1})$ has the same set of values to its arguments as I_2 in
$\phi_2 = \psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_{02}, a_{(12,j_{12})}, \mathbf{d}_{12}, a_{(22,j_{22})}, \ldots, a_{(r2,j_{r2})}, \mathbf{d}_{r2})$ but these evaluate to different

truth values when interpreted over M_2. Clearly if all the arguments of both instances were in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$, then as shown above, M would contain conflicts. (Hence P cannot be in σ.) Hence consider the case when atleast one argument of atleast one of I_1 or I_2 is in $EV(\varphi(\mathbf{x}))$.

Let the instances of P in ψ be numbered from 1 to q from left to right. Let I_1 be the m^{th} and I_2 be the n^{th} instance respectively.

Suppose $m = n$. Then I_1 and I_2 have the same arguments (and not just the same set of values to their arguments). Consider the largest index j such that $v_j \in EV(\varphi(\mathbf{x}))$ is an argument of I_1 (and hence of I_2). Since $\varphi \in \mathbf{QPS}_\sigma(\Sigma)$, v_j is the highest indexed variable of I_1 and I_2. Now let the common value of v_j in ϕ_1 and ϕ_2 be $a_{j,u}$. Then while we were constructing the interpretations in M_2 of all the predicates by moving left to right in the quantifier prefix sequence of φ, as soon as the value for v_j is substituted, the values for all the arguments of I_1 (and I_2) would have been specified (since v_j is the highest indexed variable of I_1 and I_2). Then in constructing ϕ_1 and ϕ_2, the configuration of the states of all predicate instances upon substitution of $a_{j,u}$ for v_j is u. Since u is a configuration, it defines exactly one state for all predicate instances and in particular for the m^{th} instance, it defines exactly one truth value. Then, both I_1 and I_2 must have the same truth value for the values of their arguments in ϕ_1 and ϕ_2 when interpreted in M_2. But this is a contradiction.

Suppose $m \neq n$. Then since $\varphi \in \mathbf{QPS}_\sigma(\Sigma)$ we have the following cases:

1. There is a $v_j \in EV(\varphi(\mathbf{x}))$ which (w.l.o.g) appears as the i^{th} argument of I_1 while the i^{th} argument v of I_2 is in $V(\varphi(\mathbf{x})) \cup EV(\varphi(\mathbf{x}))$. Then the value substituted for v_j is $a_{j,u}$ and for v is either some b value or some $a_{t,w} (t \neq j)$. Either of the latter 2 values of v are different from $a_{j,u}$. Clearly I_1 and I_2 then do not have the same set of values to their arguments - a contradiction.

2. I_1 and I_2 have the same argument lists. Then consider the instance I_1 appearing in ϕ_2. Since I_1 and I_2 have the same argument lists, they have the same values to their arguments in ϕ_2. But I_1 in ϕ_1 and I_2 (in ϕ_2) have the same values to their arguments. So I_1 in ϕ_1 and I_1 in ϕ_2 have the same argument lists. Now suppose (w.l.o.g) I_1 in ϕ_1 is **True** while I_2 in ϕ_2 is **False** when interpreted in M_2. Then since there are no internal conflicts, I_1 in ϕ_2 is also **False** when interpreted in M_2. Thus I_1 in ϕ_1 conflicts with I_1 in ϕ_2 - which, as shown above, is not possible.

Thus there are no external conflicts as well.

Thus the interpretations of the predicates of $\Sigma \setminus \sigma$ in M_2 are conflict-free. Note that since the σ predicates have only arguments of kind (A) described above, they are defined in M_2 exactly as they are defined in M restricted to A_2 (and hence are also conflict-free). Thus if M_2' is the sub-structure of M generated by A_2, then $M_2'|_\sigma = M_2|_\sigma$.

Now note that by our construction, all the predicate instances, when interpreted in M_2 for the values of their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, a_{(1,j_1)}, \mathbf{d}_1, a_{(2,j_2)}, \ldots, a_{(r,j_r)}, \mathbf{d}_r)$ have the same states (i.e. truth values) as they have when interpreted in M, for the values of their arguments in $\psi(\mathbf{b}_1, \mathbf{b}_2, \mathbf{d}_0, e_1, \mathbf{d}_1, e_2, \ldots, e_r, \mathbf{d}_r)$. Since $M \models \varphi(\mathbf{b}_1)$, the latter evaluates to **True** and hence the former also evaluates to **True**. In other words, $M_2 \models \varphi(\mathbf{b}_1)$.

From the preceding two paragraphs, $\varphi(\mathbf{x}) \in \mathbf{EBS}_\Sigma(\sigma)$.

Closure Properties of $\mathcal{QPS}$

The closure properties of $\mathcal{QPS}$ are the same as given in the paper. Only their proofs are slightly modified due to the modified definition of the **QPS** property.
Let $\varphi_i(\mathbf{x}_i) \in \mathcal{QPS}$. Then $\varphi_i(\mathbf{x}_i) \in \mathbf{QPS}_{\Sigma_i}(\sigma_i)$ for some $\sigma_i \in \Sigma_i$ where $\Sigma_i = Sig(\varphi_i)$. Let $\mathcal{C} = (\Sigma_1 \cap \Sigma_2 = \sigma_1 \cap \sigma_2)$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$ and $\sigma = \sigma_1 \cup \sigma_2$.

1. Closure under $\oplus$ ($\oplus = \wedge, \vee$)
   Let $\varphi(\mathbf{x}) = \varphi_1(\mathbf{x}_1) \oplus \varphi_2(\mathbf{x}_2)$. Firstly assume that there are no primed variables in either φ_1 or φ_2. Then rename each quantified variable v of φ_2 to v'. Let this new formula be $\varphi_2'(\mathbf{x}_2)$. Now bring φ in prenex normal form with the leftmost $\exists$ quantified variables of φ being the leftmost $\exists$ quantified variables of φ_1 and φ_2'.

   Then $V(\varphi(\mathbf{x})) = V(\varphi_1(\mathbf{x}_1)) \cup V(\varphi_2'(\mathbf{x}_2))$, $AV(\varphi(\mathbf{x})) = AV(\varphi_1(\mathbf{x}_1)) \cup AV(\varphi_2'(\mathbf{x}_2))$, $EV(\varphi(\mathbf{x})) = EV(\varphi_1(\mathbf{x}_1)) \cup EV(\varphi_2'(\mathbf{x}_2))$.

   Consider $P \in \sigma$ of arity ≥ 1. Then we have the following cases:

   (a) $P \in \sigma_1 \cap \sigma_2$. Then, an instance of P in φ is either an instance of P in φ_1 or φ_2'. In the former case, all its arguments are in $V(\varphi_1(\mathbf{x}_1)) \cup AV(\varphi_1(\mathbf{x}_1))$ and hence in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$. In the latter case, all its arguments are in $V(\varphi_2'(\mathbf{x}_2)) \cup AV(\varphi_2'(\mathbf{x}_2))$ and hence in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$.

   (b) $P \in \sigma_1 \setminus \sigma_2$ or $P \in \sigma_2 \setminus \sigma_1$. Now since $\mathcal{C}$ holds, $(\sigma_1 \setminus \sigma_2) \cap \Sigma_2 = \emptyset$ and $(\sigma_2 \setminus \sigma_1) \cap \Sigma_1 = \emptyset$. Then for $P \in \sigma_1 \setminus \sigma_2$, every instance of P in φ is an instance of P in φ_1. Then all its arguments are in $V(\varphi_1(\mathbf{x}_1)) \cup AV(\varphi_1(\mathbf{x}_1))$ and hence in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$. Similarly, for $P \in \sigma_2 \setminus \sigma_1$, for any instance of P, all its arguments are in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$.

   Consider now $P \in \Sigma \setminus \sigma$. Again since $\mathcal{C}$ holds, $(\Sigma_i \setminus \sigma) \subseteq (\Sigma_i \setminus \Sigma_j)$, $\;i, j \in \{1, 2\}, i \neq j$. Then consider $P \in \Sigma \setminus \sigma$. Then $P \in \Sigma_1 \setminus \sigma$ or $P \in \Sigma_2 \setminus \sigma$. Then either (a) $P \notin \Sigma_2$ and $P \in \Sigma_1 \setminus \sigma_1$ and hence every instance of P appears only in φ_1 or (b) $P \notin \Sigma_1$ and $P \in \Sigma_2 \setminus \sigma_2$ and hence every instance of P appears only in φ_2.

   In either case, it is easy to see that the **QPS** conditions are satisfied by P.

   Thus $\varphi(\mathbf{x}) \in \mathbf{QPS}_\Sigma(\sigma) \subseteq \mathcal{QPS}$.

2. $\neg$– Non-closure
   $\mathcal{QPS}$ is also not closed under negation. Consider $\varphi = \exists x \forall y (P(x, x) \wedge P(y, y))$. Clearly $\varphi \in \mathcal{QPS}$. But in $\neg\varphi = \forall x \exists y \neg(P(x, x) \wedge P(y, y))$, the *exists* quantified y as the 1^{st} argument of $P(y, y)$ comes after the $\forall$ quantified x in the 1^{st} argument of $P(x, x)$ in the prefix structure of $\neg\varphi$.Hence $\neg\varphi \notin \mathcal{QPS}$.

3. $\exists-$ closure.
   Under $\exists$ quantification, we note that the sets $V(\cdot), AV(\cdot)$ and $EV(\cdot)$ remain unchanged and the highest indexed argument of any predicate instance continues to remain its highest indexed argument. Hence $\mathcal{QPS}$ is closed under $\exists$ quantification.

4. $\forall-$ Non-closure
   Consider the formula $\varphi(x) = \exists y(P(y) \vee P(x))$. Clearly $\varphi \in \mathbf{QPS}_{\{P\}}(\emptyset)$. But $\forall x \varphi(x) \notin \mathbf{QPS}_{\Sigma}(\sigma)$ for any $\sigma \subseteq \Sigma$.

   Consider the sub-class of $\mathcal{QPS}$ in which each formula $\varphi(\mathbf{x})$ is such that if for the variable z being $\forall$ quantified, z occurs as the i^{th} argument of a predicate P, then the i^{th} argument in each instance of P is in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$. Then under $\forall$ quantification of z, $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$ remains unchanged and the i^{th} argument of each instance of P continues to be in $V(\varphi(\mathbf{x})) \cup AV(\varphi(\mathbf{x}))$. Its easy to check that the $\forall-$ quantified formula is in $\mathcal{QPS}$. (More precisely, if $\varphi(z, \mathbf{x}) \in \mathbf{QPS}_{\Sigma}(\sigma)$, then $\forall z \varphi(z, \mathbf{x}) \in \mathbf{QPS}_{\Sigma}(\sigma)$.) Hence this sub-class is closed under $\forall$.

11.2 $\mathcal{EBS}$ and semi-linear sets

We first introduce some notation. Let V be the set of all vocabularies. Let $\mathcal{EBS} = \bigcup_{\Sigma \in V} \bigcup_{\sigma \subseteq \Sigma} \mathbf{EBS}_{\Sigma}(\sigma)$. We look now at the relation between $\mathcal{EBS}$ and semi-linear sets. Before that, we need the following result.

Claim 6:

A sentence $\varphi \in \mathcal{EBS}$ iff either φ is unsatisfiable or there exists $\mathcal{B}'$ such that φ has a model of every cardinality ranging from $\mathcal{B}'$ to the cardinality of the φ's largest model.

Proof:

A formula $\psi \in \mathbf{EBS}_{\Sigma}(\emptyset)$ iff $\Sigma = Sig(\psi)$ and there exists $\mathcal{B}$ such that for every $\Sigma-$structure M having universe A with $|A| \geq \mathcal{B}$ satisfying $M \models \psi$, there exists $A_1 \subseteq A$ such that

1. $|A_1| \leq \mathcal{B}$
2. For all A_2, $A_1 \subseteq A_2 \subseteq A$, there exists a $\sigma-$structure M_2 with universe $A_2 \cup C$ such that $M_2 \models \psi$. (where $C = \left\{c^M | c \text{ } constant \text{ }, c \in \sigma\right\}$ is the set of interpretations in M of the constants of σ).

Assume that $\varphi \in \mathcal{EBS}$. Then $\varphi \in \mathcal{EBS}_{\Sigma}(\emptyset)$ where $\Sigma = Sig(\varphi)$. Then either φ is unsatisfiable or there is a $\mathcal{B}$ such that for the largest model M of φ, the size of its universe A is $\geq \mathcal{B}$ and there is $A_1 \subseteq A$ satisfying the properties as memtioned above. Then take $\mathcal{B}' = \mathcal{B} + |C|$. Then for every A_2 such that $A_1 \cup C \subseteq A_2 \subseteq A$, there exista a $\Sigma-$ structure M_2 with universe A_2 satisfying $M_2 \models \varphi$. Then the 'only-if' part of the Lemma holds.

We now prove the 'if' part. Assume that the 'if' part is true for a formula φ. Let $\Sigma = Sig(\varphi)$ and consider a $\Sigma-$structure M having universe A with $|A| \geq \mathcal{B}$ where $\mathcal{B} = \mathcal{B}' + |C|$ such that $M \models \varphi$. Consider any subset A_1 of A such that $|A_1| = \mathcal{B}$. Consider A_2 such that $A_1 \subseteq A_2 \subseteq A$. Now since $C \subseteq A$, we have $A_1 \subseteq A_2 \cup C \subseteq A$. Then $\mathcal{B} \leq |A_2 \cup C| \leq$ size of φ's largest model. Then by the 'if' part, there is a model M_2' of φ of cardinality $|A_2 \cup C|$. Then consider the model M_2 with universe $A_2 \cup C$ and isomorphic to M_2'. Since boolean queries are closed under isomorphism [LL98], $M_2 \models \varphi$. Thus $\varphi \in \mathbf{EBS}_\Sigma(\emptyset)$ and hence $\varphi \in \mathcal{EBS}$.

We now look at the relation between $\mathcal{EBS}$ and semi-linear sets.

By **Claim 6**, if $\varphi \in \mathcal{EBS}$, either φ is unsatisfiable or there is a $\mathcal{B}'$ such that φ has models of all cardinalities ranging from $\mathcal{B}'$ to the cardinality of φ's largest model. The *spectrum* of a formula φ is defined as the set of the cardinalities of all its models. Let S_φ be the spectrum of φ restricted to countable cardinalities. Then for $\varphi \in \mathcal{EBS}, S_\varphi$ is of the form

1. $S_\varphi = \{\mathcal{B}_{01}, \mathcal{B}_{02}, \ldots, \mathcal{B}_{0n}\}$ if φ has no infinite model. Here $\mathcal{B}_{0i} < \mathcal{B}_{0(i+1)}$.

2. $S_\varphi = \{\mathcal{B}_{01}, \mathcal{B}_{02}, \ldots, \mathcal{B}_{0n}\} \cup \{\mathcal{B}', \mathcal{B}'+1, \mathcal{B}'+2, \ldots, \}$ if φ has an infinite model. Here $\mathcal{B}_{0i} < \mathcal{B}_{0(i+1)}$, $\mathcal{B}_{0n} < \mathcal{B}'$.

A *linear* set produced by the pair $(p, q), p, q \in \aleph$ is defined as $\{p + iq | i \in \aleph\}$. A *semi-linear* set is a finite union of linear sets. Then we see that S_φ is a finite union of linear sets produced by the pairs (a) $(\mathcal{B}_{0i}, 0),\ 1 \leq i \leq n$ for (1) and (b) $(\mathcal{B}_{0i}, 0),\ 1 \leq i \leq n$ and $(\mathcal{B}', 1)$ for (2). Thus for $\varphi \in \mathcal{EBS}$, S_φ is a semilinear set.

What can we say about the converse? That is, given a semi-linear set S, is there $\varphi \in \mathcal{EBS}$ such that $S_\varphi = S$? Clearly, if it is, then as seen above it is a semi-linear set with q being 0 or 1. But q, for a general semi-linear set is not restricted to be 0 or 1. So for the converse to also be true, we must consider a class larger than $\mathcal{EBS}$. We then relax the definition of the **EBS** property as below:

A formula $\varphi(\mathbf{x})$ is said to satisfy the **Extensible Bounded Submodel** property with respect to $\sigma \subseteq \Sigma$ (denoted as $\varphi(\mathbf{x}) \in \mathbf{EBS}_\Sigma(\sigma)$) if $Sig(\varphi) = \Sigma$ and there exists a cardinal $\mathcal{B}$ (dependent on φ and σ in general) such that for every $\Sigma-$structure M having universe A with $|A| \geq \mathcal{B}$ and for every $\mathbf{a} \in A^{|\mathbf{x}|}$, if $M \models \varphi(\mathbf{a})$ then there exists $A_1 \subseteq A$ such that

1. $\mathbf{a} \in A_1^{|\mathbf{x}|}$

2. $|A_1| \leq \mathcal{B}$

3. There exists a recursive function $f_\varphi : \aleph \rightarrow S_\varphi$ with the property that $f_\varphi(x)$ is the smallest number in S_φ greater than x. Then

(a) for all A_2 such that $A_1 \subseteq A_2 \subseteq A$ and $|A_2| < \aleph$ there exists A'_2 such that $A_2 \subseteq A'_2 \subseteq A$, $|A'_2 \cup C| = f_\varphi(|A_2|)$ (C is the set of interpretations of the constants of Σ) and if M'_2 is the substructure of M generated by A'_2, then there exists a $\Sigma-$structure M_2 such that $M_2|_\sigma = M'_2|_\sigma$ and $M_2 \models \varphi(\mathbf{a})$.

(b) for all A_2 such that $A_1 \subseteq A_2 \subseteq A$ and cardinality of A_2 is infinite, if M'_2 is the substructure of M generated by A_2, then there exists a $\Sigma-$structure M_2 such that $M_2|_\sigma = M'_2|_\sigma$ and $M_2 \models \varphi(\mathbf{a})$.

Note the following:

- By the above definition, f_φ is a monotone function.
- The size of S_φ must be infinite. Suppose it was finite, it would have a largest element say t. Then $f_\varphi(t)$ will not be defined.
- If the largest model of φ has finite size, then there cannot exist any f_φ satisfying the above conditions. Suppose it existed. Let A be the universe of the model. Then choosing $A_2 = A$, there must exist $A'_2 \subseteq A$ such that $|A'_2 \cup C| = f_\varphi(|A|) > |A|$ and satisfying the conditions of 3a. But since $A'_2 \cup C \subseteq A$, we have a contradiction.

Now, since the motivation behind this relaxation is to achieve the equivalence that a sentence $\varphi \in \mathcal{EBS}$ iff S_φ is semi-linear, we might have to impose some additional constraints on f_φ. We now investigate what these additional constraints are.

We firstly have the following.

Claim 7:
If a sentence $\varphi \in \mathcal{EBS}$, then

1. φ is unsatisfiable or the largest model of φ is of finite size or
2. For some $n \geq 0$, $S_\varphi = \{\mathcal{B}_{01}, \mathcal{B}_{02}, \ldots, \mathcal{B}_{0n}\} \cup \left\{ f_\varphi^{(i)}(\mathcal{B}) | i \geq 1 \right\}$.

Proof:
(1) is easy to see. For (2), since $\varphi \in \mathcal{EBS}, \varphi \in \mathbf{EBS}_\Sigma(\sigma)$. Then clearly there exist models of sizes $f_\varphi^{(i)}(\mathcal{B}), i \geq 1$. Further by the property that $f_\varphi(x)$ is the smallest number in S_φ greater than x, we see that there are no models of sizes in between $f_\varphi^{(i)}(\mathcal{B})$ and $f_\varphi^{(i+1)}(\mathcal{B})$ for all $i \geq 1$.

We recall that the modified definition of the **EBS** property was to achieve the equivalence that a formula $\varphi \in \mathcal{EBS}$ iff S_φ is semi-linear. Then we need to know what are functions f_φ such that S_φ is semi-linear.

A property of semi-linear sets
We state the following without proof.

Claim 8:
Consider a set $S \subseteq \aleph$. Consider the sequence s of elements of S ordered by $<$. Then consider the 'difference' sequence d of S which is formed by taking the difference of consecutive elements of s. (Eg. if $s = \{1, 3, 7, 14, \ldots\}$, $d = 2, 4, 7, \ldots$).
Then S is semi-linear iff d is of the form

$$a_1, a_2, a_3, \ldots, a_n(b_1, b_2, \ldots, b_m)^*$$

(* means repetition infinite number of times)

Then if S_φ is semi-linear we have the following:

1. Either it is finite or
2. For some $n_0 \geq 0$,

$$f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B}) = b_j \quad i = n_0 + j + km, 1 \leq j \leq m, k = 0, 1, \ldots$$

   Thus for $i > n_0$, the difference $\mathcal{D}(i) = f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B})$ is a periodic function.

(Note that if S_φ was semi-linear with the generating pairs of the form $(p, 0)$ and atleast one pair $(p, 1)$, then the difference sequence of D is of the form $a_1, a_2, \ldots, a_n(1)^*$. Then $\mathcal{D}(i) = f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B}) = 1$ for $i > n_0$. Then $f_\varphi(x) = x$ for all $x \geq f_\varphi^{n_0}(\mathcal{B})$. We can see that this special case is indeed the old definition of **EBS** with the cardinal $f_\varphi^{n_0}(\mathcal{B})$ being taken as the cardinal satisfying the **EBS** conditions.)

We now add this condition to the relaxed **EBS** definition and state the modified definition below.

A formula $\varphi(\mathbf{x})$ is said to satisfy the **Extensible Bounded Submodel** property with respect to $\sigma \subseteq \Sigma$ (denoted as $\varphi(\mathbf{x}) \in \mathbf{EBS}_\Sigma(\sigma)$) if $Sig(\varphi) = \Sigma$ and there exists a cardinal $\mathcal{B}$ (dependent on φ and σ in general) such that for every $\Sigma-$structure M having universe A with $|A| \geq \mathcal{B}$ and for every $\mathbf{a} \in A^{|\mathbf{x}|}$, if $M \models \varphi(\mathbf{a})$ then there exists $A_1 \subseteq A$ such that

1. $\mathbf{a} \in A_1^{|\mathbf{x}|}$
2. $|A_1| \leq \mathcal{B}$
3. There exists a recursive function $f_\varphi : \aleph \rightarrow S_\varphi$ with the property that (a) $f_\varphi(x)$ is the smallest number in S_φ greater than x (b) there exists some n_0 such that for $i > n_0$, the difference $\mathcal{D}(i) = f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B})$ is a periodic function. Then
   (a) for all A_2 such that $A_1 \subseteq A_2 \subseteq A$ and $|A_2| < \aleph$ there exists A'_2 such that $A_2 \subseteq A'_2 \subseteq A$, $|A'_2 \cup C| = f_\varphi(|A_2|)$ (C is the set of interpretations of the constants of Σ) and if M'_2 is the substructure of M generated by A'_2, then there exists a $\sigma-$structure M_2 such that $M_2|_\sigma = M'_2|_\sigma$ and $M_2 \models \varphi(\mathbf{a})$.

(b) for all A_2 such that $A_1 \subseteq A_2 \subseteq A$ and cardinality of A_2 is infinite, if M'_2 is the substructure of M generated by A_2, then there exists a $\sigma-$structure M_2 such that $M_2|_\sigma = M'_2|_\sigma$ and $M_2 \models \varphi(\mathbf{a})$.

As before, let $\mathcal{EBS} = \bigcup_{\Sigma\in V}\bigcup_{\sigma\subseteq\Sigma}\mathbf{EBS}_\Sigma(\sigma)$. Again as before, we have $\mathbf{EBS}_\Sigma(\sigma_1) \subseteq \mathbf{EBS}_\Sigma(\sigma_2)$ if $\sigma_2 \subseteq \sigma_1 \subseteq \Sigma$. Hence $\mathcal{EBS} = \bigcup_{\Sigma\in V}\mathbf{EBS}_\Sigma(\emptyset)$.

We now prove that $\varphi \in \mathcal{EBS}$ iff S_φ is semi-linear.

Proof:

Let S_φ be semi-linear, then (a) either it is finite or (b) its difference sequence is of the form $a_1, \ldots, a_n(b_1, \ldots, b_m)^*$. If it is finite, then take $\mathcal{B}$ to be one more than the size of the largest model of φ. Clearly then $\varphi \in \mathcal{EBS}$. Else consider the function $f_\varphi : \aleph \to S_\varphi$ defined as below. Let $\mathcal{B}_O$ be the size of the smallest model of φ, $X(j) = \Sigma_{i=1}^{i=j} a_i + \mathcal{B}_0$, $C_1 = X(n)$, $Y(j) = \Sigma_{i=1}^{i=j} b_i$, $C_2 = Y(m)$ and $Z(k,j) = Y(j) + C_1 + k.C_2$.

$$f_\varphi(x) \quad = \quad \begin{cases} X(0) & x \le \mathcal{B}_0 \\ X(k) & X(k-1) < x \le X(k), 1 \le k \le n \\ Z(k,j) & Z(k,j-1) < x \le Z(k,j), 1 \le j \le m, k \ge 0 \end{cases}$$

Let $\mathcal{B} = |\mathbf{x}|$ (No. of free vars of $\varphi(\mathbf{x})$.) Its easy to see that f_φ is recursive and satisfies the following:

1. $f_\varphi(x)$ is the smallest element of S_φ which is greater than or equal to x.

2. For some $r \ge 0$, $S_\varphi = \{\mathcal{B}_{01}, \ldots, \mathcal{B}_{0r}\} \cup \{f_\varphi^{(i)}(\mathcal{B}) | i \ge 1\}$.

3. Let n_0 be the smallest $i-1$ such that $f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B}) = b_1$. Then

$$f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B}) = b_j \quad i = n_0 + j + km, 1 \le j \le m, k = 0, 1, \ldots$$

   Thus for $i > n_0$, the difference $\mathcal{D}(i) = f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B})$ is a periodic function.

Then following the proof of **Claim 6**, its easy to show that $\varphi \in \mathbf{EBS}_\Sigma(\emptyset) \subseteq \mathcal{EBS}$.

Conversely assume $\varphi \in \mathcal{EBS}$. Then $\varphi \in \mathbf{EBS}_\Sigma(\sigma)$ for some $\sigma \in \Sigma$. Then for some $r \ge 0$, $S_\varphi = \{\mathcal{B}_{01}, \ldots, \mathcal{B}_{0r}\} \cup \{f_\varphi^{(i)}(\mathcal{B}) | i \ge 1\}$. Now since for some $n_0 \ge 0$, for all $i > n_0$, $\mathcal{D}(i) = f_\varphi^{(i)}(\mathcal{B}) - f_\varphi^{(i-1)}(\mathcal{B})$ is a periodic function, we have that the difference sequence of S_φ is of the form $a_1, \ldots, a_n(b_1, \ldots, b_m)^*$. Then S_φ is semi-linear.

Ofcourse with the last definition of the **EBS** property, the closure properties are not guaranteed!

Chapter 12

Conclusion and Future Work

We looked at graph transformation systems in which each state is a typed graph and transitions from one state to another involve graph transformations. We are then interested in knowing whether certain safety properties are true in all the reachable states of the system given a set of initial states. This can be translated into the problem of reachability of states violating the safety property from the initial set of states. We illustrated this with the particular example of the library system which can be modeled as a graph transforming system. We then introduced the safety property of checking whether there exists a book on shelf yet there is a pending claim for the title of the book.

We then looked at how the problem of reachability analysis is undecidable in even infinite state graph transforming systems. (The problem is known to be undecidable in infinite state systems in general). We then developed a procedure `InductionSolve` which uses a combination of inductive reasoning and bounded model checking to perform reachability analysis in infinite state systems in general. (The procedure though is not guaranteed to terminate on all inputs.) We then gave a logical formalism for graph transforming systems where a graph transforming system (GTS) is expressed using a set of operations and a set of sub-actions on nodes and edges. We considered a special kind of sub-actions and posed the question that whether the transition relation $\mathcal{T}(s, s')$ on states defined by an arbitrary GTS can also be defined by a GTS using the special sub-actions. The question though still open, we showed that for GTSes satisfying a certain condition, it is indeed possible to construct a GTS using the special sub-actions which defines the same transition relation on states.

We looked at the procedure `InductionSolve` which uses a validity checker and found that there are are two ways of non-termination of the procedure - one by an unbounded increase in its iterator, which cannot be avoided in general and the other by the non-termination of its validity checker. We saw that the bounded model property for the formulae of `InductionSolve` enables us to decide their validity and thus enables us to construct a terminating validity checker. We looked at 2 classes of GTSes which have bounded models for $\mathcal{T}(s, s')$ and showed that the library system is indeed in one of these classes. We finally considered certain safety properties and initial conditions which have the bounded model property and then showed how this can be used to get bounded models for the formulae of `InductionSolve` and computed expressions for the bounds. We used this to construct a decision procedure (validity checker) for the formulae of `InductionSolve`. Finally we showed that in the library system, for the particular safety property mentioned above, we can construct a decision procedure for it and find that `InductionSolve` too terminates for it and proves that the library system indeed satisfies the safety property.

Some of the future work is mentioned below:

1. We firstly note that the results in chapter 9 talk about GTSes satisfying certain *semantic* conditions and mention one example for each, each example satisfying certain *syntactic* properties which ensure the semantic conditions. We would firstly like to replace the semantic conditions by syntactic conditions while still preserving the generality of the semantic conditions. It would be even better to get a syntactic characterization which is more general than the semantic conditions. However, if we are unable to do either of these, we would atleast like to get more examples satisfying syntactic properties which ensure the semantic conditions too. It would be even better if we could, just as it was done for the classical decision problem (of checking whether an arbitrary FO formula is satisfiable), classify all the formulae on syntactic criteria into 2 sets of classes - one set containing classes for which we can *decide* whether formulae of the class satisfy the semantic conditions and the other set containing classes for which the problem of checking whether a formula of the class satisfies the semantic conditions is undecidable.

2. We would like to know how expressive the formalism for graph transformation systems introduced in Chapter 5 is.

3. We would like to know whether the transition relation captured by an arbitrary GTS can be captured by a GTS using the special sub-actions mentioned in Chapter 6.

4. The concept of Ehrenfeucht-Fra$\ddot{i}$ss$\acute{e}$ games is very relevant to our problem. We briefly this concept is as follows:

   The classical Ehrenfeucht-Fra$\ddot{i}$ss$\acute{e}$ game on two $\sigma-$structures $\mathcal{A}$ and $\mathcal{B}$ consists of two players - a spoiler and a duplicator. The game is played in rounds and in each round there are two moves - one by the spoiler and the other by the duplicator. The spoiler selects one of the structures and chooses an element from it. The duplicator then chooses an element from the other structure. The objective of the duplicator is to show that $\mathcal{A}$ and $\mathcal{B}$ are isomorphic while that of the spoiler is to show they are not. More precisely, in a k round Ehrenfeucht-Fra$\ddot{i}$ss$\acute{e}$ game, the duplicator tries to show a *partial isomorphism* between $\mathcal{A}$ and $\mathcal{B}$. This is the winning condition for the duplicator. If there is a strategy for the duplicator to show a partial isomorphism regardless of the spoiler's moves, it is denoted as $\mathcal{A} \equiv_k \mathcal{B}$. The theorem by Ehrenfeucht and

Fraïssé says that $\mathcal{A} \equiv_k \mathcal{B}$ iff $\mathcal{A}$ and $\mathcal{B}$ agree on all FO sentences of quantifier rank atmost k.

We then wish to identify classes of σ_S-structures (i.e. states as defined in Chapter 5) such that each structure is $\equiv_k$ to some structure of bounded size. We then identify the properties of these classes of structures which are expressible in FO. Then we are guaranteed by the Ehrenfeucht and Fraïssé theorem that the properties are true in structures of these classes iff they are true in structures of bounded size. In other words, these properties will have bounded models. What are such classes of structures and what are the properties is something we wish to explore.

5. While bounded model property is one way of deciding the formulae $IS1(k)$ and $IS2(k)$, we would like to explore other ways of deciding these formulae.

References

[LL98] Leonid Libkin, *Elements of Finite Model Theory*, Springer Verlag, 1998.

[BGG97] E. Börger, E. Grädel and Y. Gurevich, *The Classical Decision Problem.*, Springer Verlag, 1997.

[HU] J. Hopcroft, R. Motwani, J. Ullman, *Introduction to Automata Theory, Languages and Computation*, Pearson Education, 2001.

[NS00] Natarajan Shankar, *Combining Theorem Proving and Model Checking through Symbolic Analysis* in Proc. of 11th Intl. Conf. on Concurrency Theory (CONCUR 2000), Lecture Notes in Computer Science series, Vol. 1877, Chp: p 1, Springer Verlag, 2000

[MSG00] M. Sheeran, S. Singh, G. Stalmarck, *Checking Safety Properties Using Induction and a SAT-Solver* in *Proc. FMCAD '00*, pp.108–125, Springer Verlag, London, UK, 2000

[BCC99] A. Biere, A. Cimatti, E. M. Clarke, et.al *Symbolic model checking using SAT procedures instead of BDDs*, *Proc. DAC'99*, pp.317–320, ACM Press, New York, USA.

[WOL87] P. Wolper, *On the relation of programs and computations to models of temporal logic*, In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Temporal Logic in Specification*, pp. 75123, Altrincham, UK, 1987. Springer-Verlag.

[KLM97] K. L. McMillan, *Symbolic model checking - an approach to the state explosion problem*, PhD thesis, SCS, Carnegie Mellon University, 1992.

[MP86] M. Y. Vardi, P. Wolper, *An automata-theoretic approach to automatic program verification*, *Proc. First IEEE Symp. on Logic in Computer Science*, pp. 322-331, 1986.

[MS94] Z. Manna and the STeP group, *STeP: The Stanford Temporal Prover*, *Technical report STAN-CS-TR-94-1518*, Computer Science Department, Stanford University, July 1994. 44 pages.

[DOTY96] C. Daws, A. Olivero, S. Tripakis, S. Yovine, *The tool Kronos*, *Proc. of DIMACS/SYCON workshop on Hybrid systems III : verification and control*, pp. 208–219, Secaucus, NJ, USA, 1996.

[DPLL62] M. Davis, G.Logemann, D. Loveland, *A machine program for theorem proving*, *Communications of the ACM*, Vol. 5, pp.394-397, 1962.

[CGJ03] E. Clarke, O. Grumberg, S. Jha, et.al, *Counterexample-guided abstraction refinement for symbolic model checking*, *Journal of ACM.*, Vol.50, pp.752–794, ACM Press, New York, NY, USA, 2003.

[FQ02] C. Flanagan, S. Qadeer, *Predicate abstraction for software verification*, *Proc. POPL '02*, pp.191–202, ACM Press, New York, NY, USA, 2002.

[CL94] L. A. Crowl, T. J. LeBlanc, *Parallel programming with control abstraction*, *Journal of ACM Trans. Program. Lang. Syst.*, Vol.16, pp. 524-576, ACM Press, New York, NY, USA, 1994.

[KP00] Y. Kesten, A. Pnueli, *Control and data abstractions: The cornerstones of formal verification*, *International Journal on Software Tools for Technology Transfer (STTT)*,Vol.2, pp.328-342, 2000

[CCG02] A. Cimatti, E. M. Clarke, E. Giunchiglia, et,al., *NuSMV 2: An OpenSource Tool for Symbolic Model Checking*, *Proc. of International Conference on Computer-Aided Verification (CAV 2002)*, pp.27-31, Copenhagen, Denmark, July 2002.

[CMR97] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, M. Lowe, *Algebraic approaches to graph transformation I: Basic concepts and double pushout approach*, *Handbook of Graph Grammars and Computing by Graph Transformation, Vol. I: Foundations (World Scientific, 1997)*, pp.163245.

[L93] M. Lowe, *Algebraic approach to single-pushout graph transformation*, *Theoretical Computer Science*, 109 (1993), pp.181224.

[LB93] Michael Lowe, Martin Beyer, *AGG - An Implementation of Algebraic Graph Rewriting*, *RTA '93: Proceedings of the 5th International Conference on Rewriting Techniques and Applications*, 1993, pp. 451-456, Springer-Verlag, London, UK.

[S97] A. Schurr, *Programmed Graph Replacement Systems*, *Handbook of graph grammars and computing by graph transformation: vol. 1: foundations*, World Scientific, Singapore, Vol.1, 1997, pp.479-546