Looking into the Crystal Ball: Requirements Evolution over Time

Alicia M. Grubb, Marsha Chechik
{amgrubb, chechik}@cs.toronto.edu

RE'16 - Sep. 15, 2016

UNIVERSITY OF TORONTO

Motivating Example

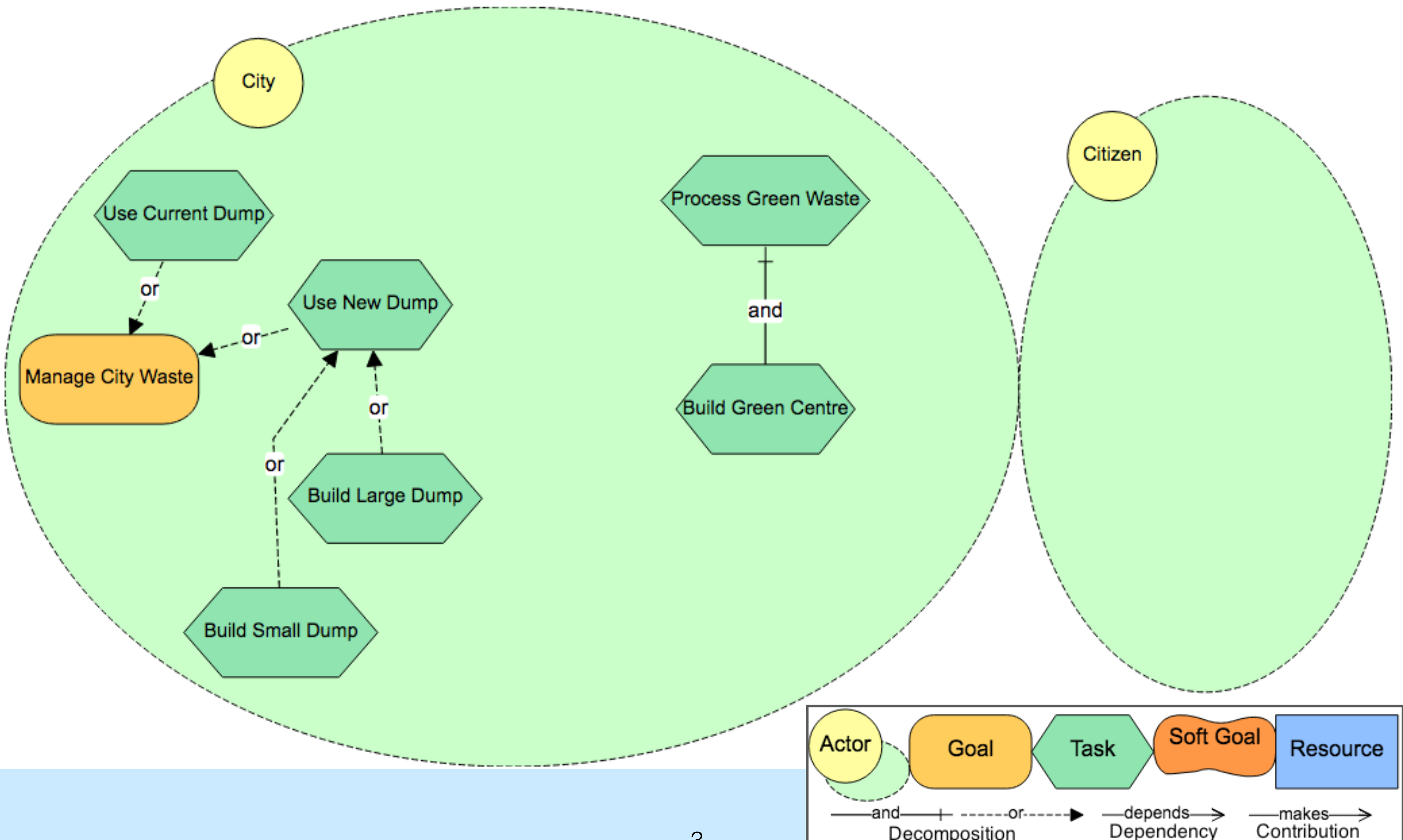
Goal: Evaluate waste management infrastructure

Intentions: Wants to be green and satisfy customer

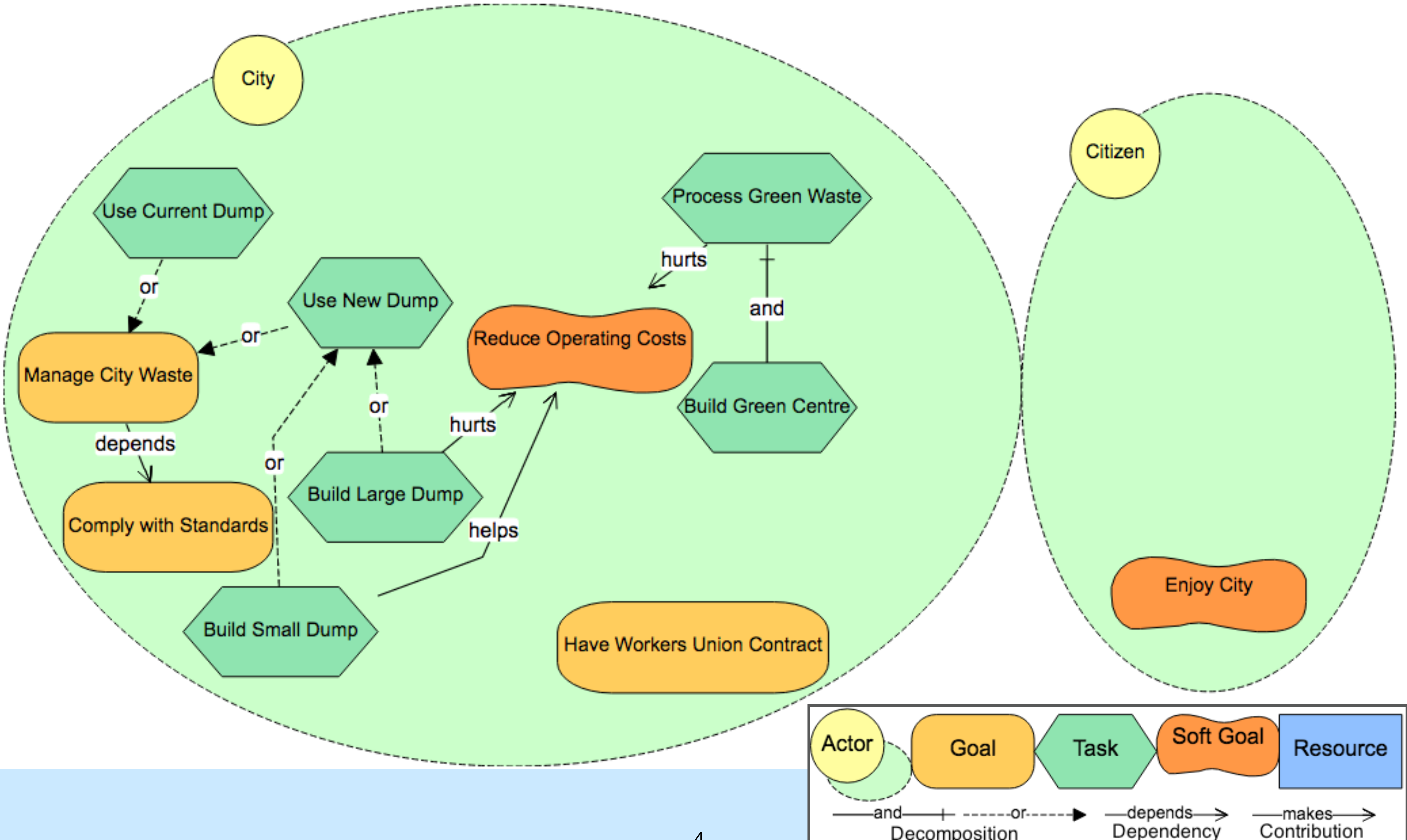
Options: Build Green Centre
Build Landfill / Dump (large, small)

Approach: Choose correct alternative(s) using goal modeling.

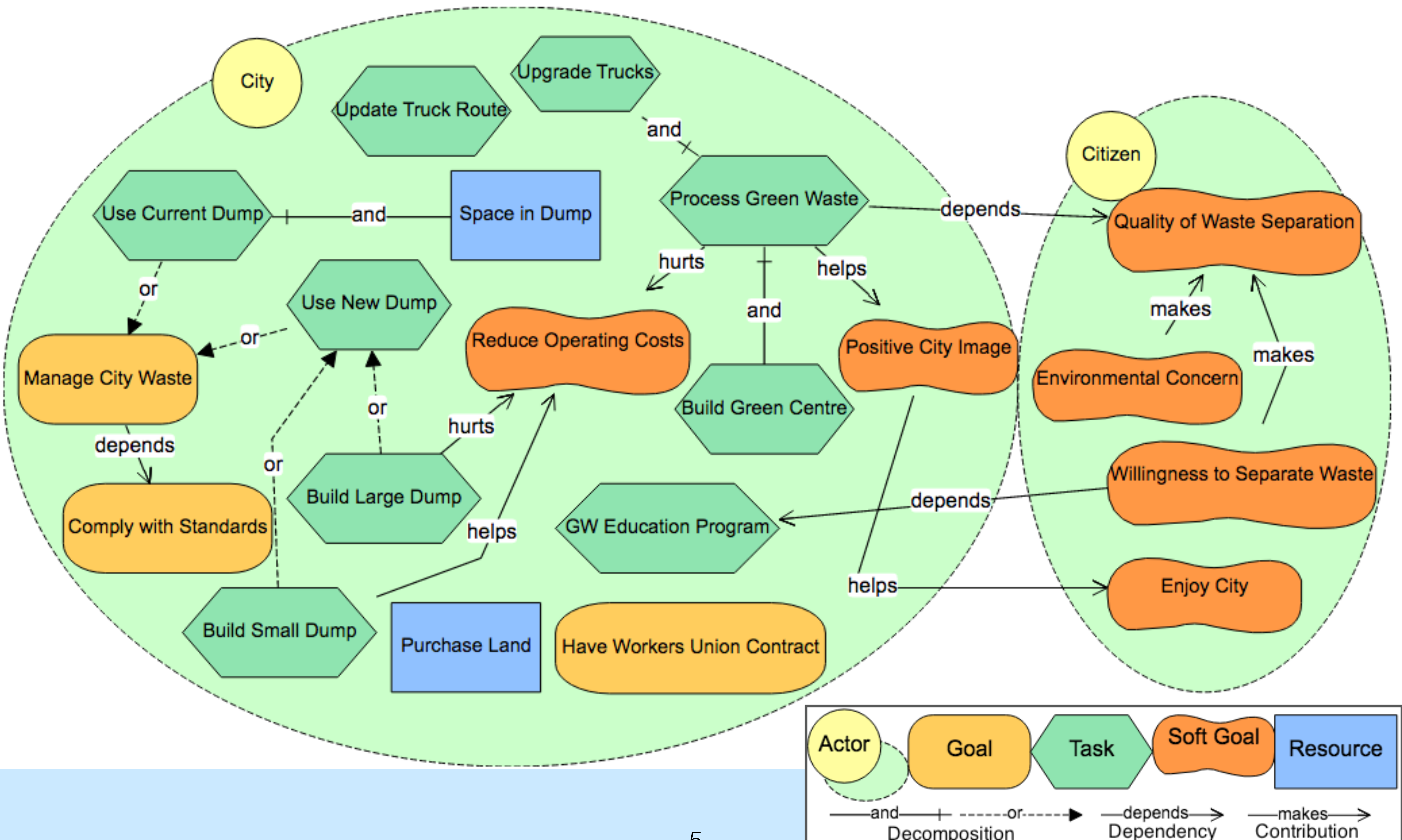
Waste Management Example



Waste Management Example



Waste Management Example



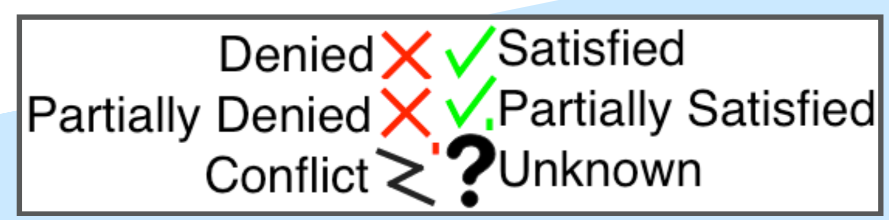
Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?

Waste Management Example

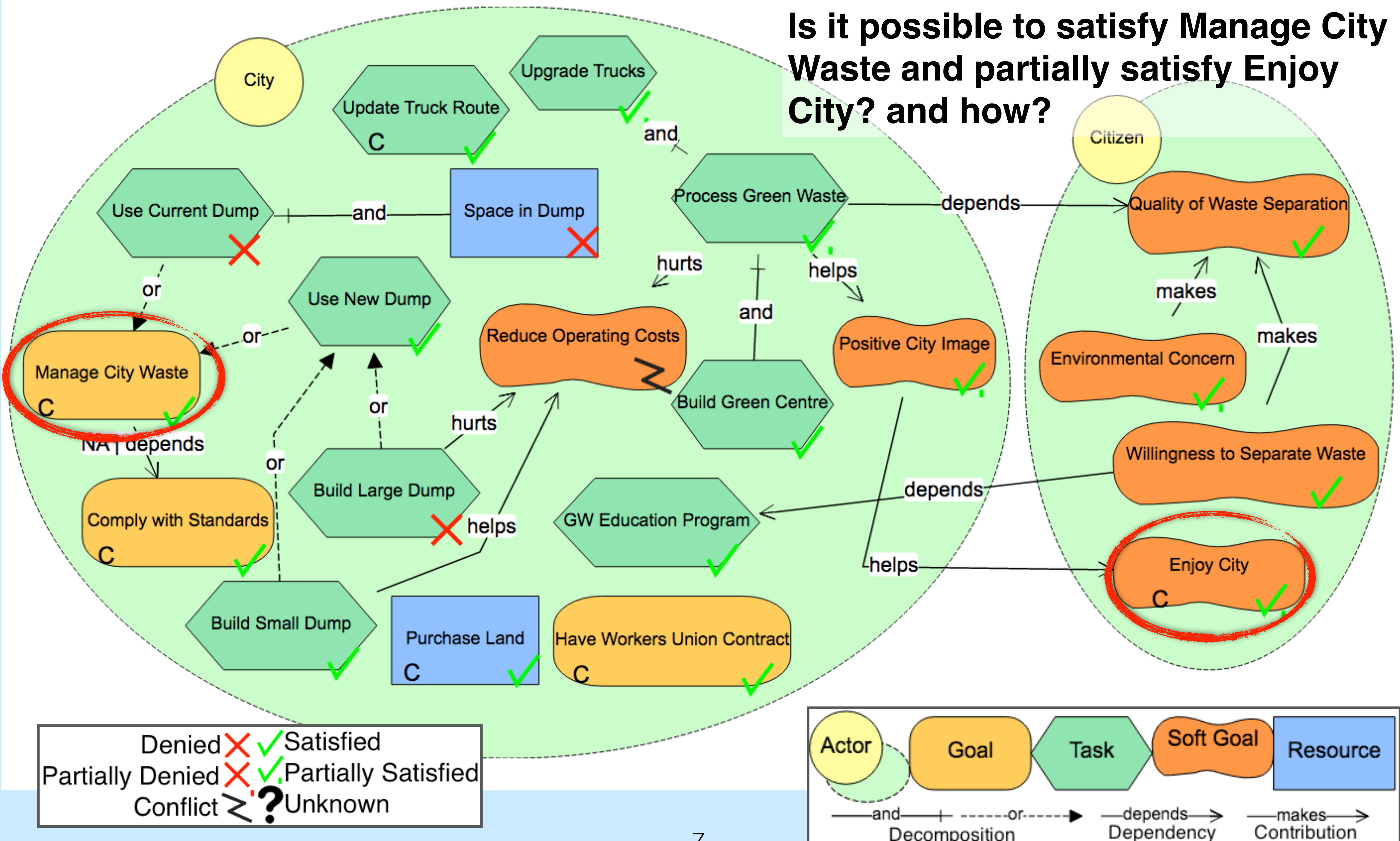
1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?

Use Qualitative Evaluation Labels with Forward Analysis and Backward Analysis



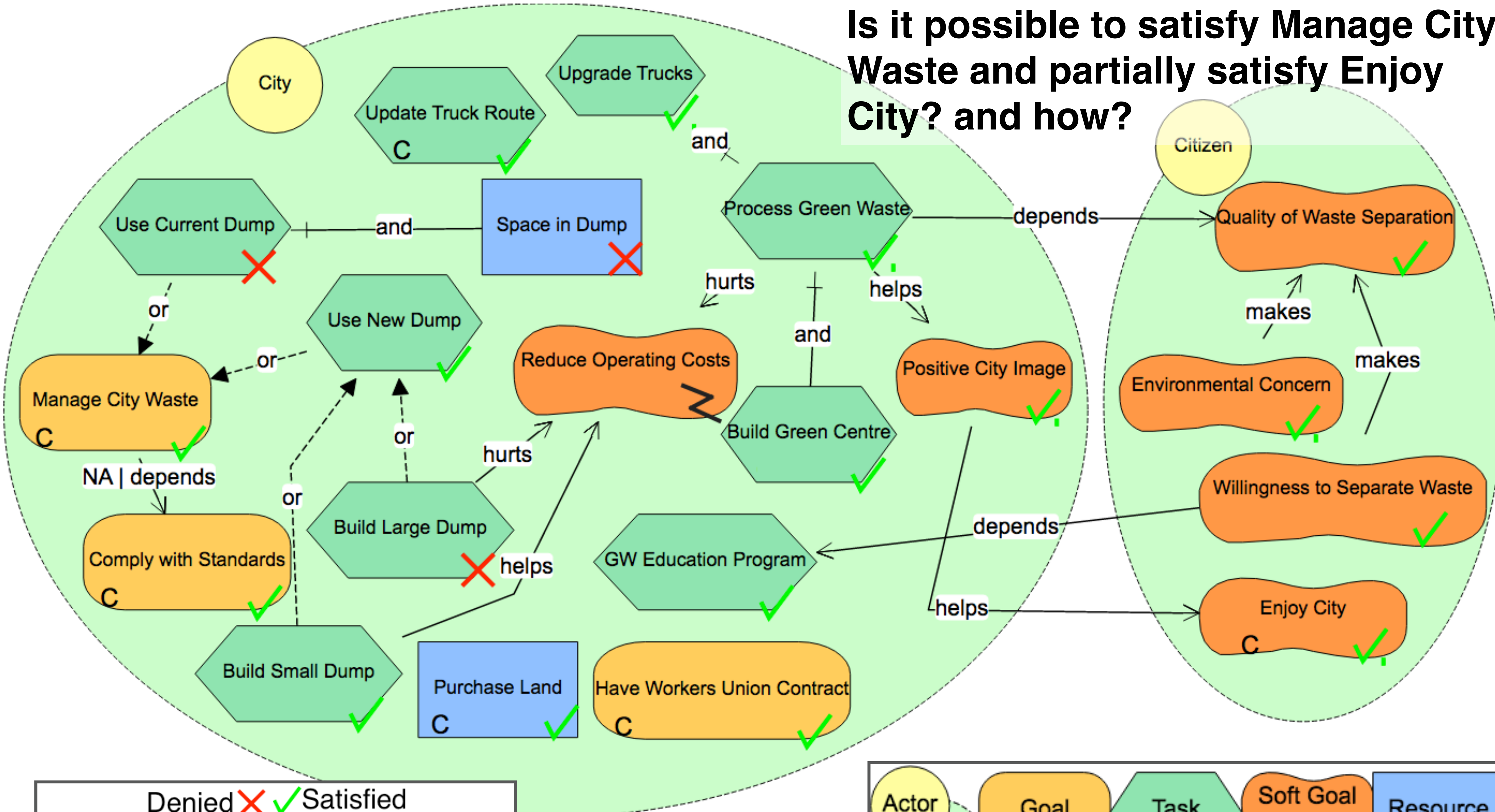
Waste Management Example

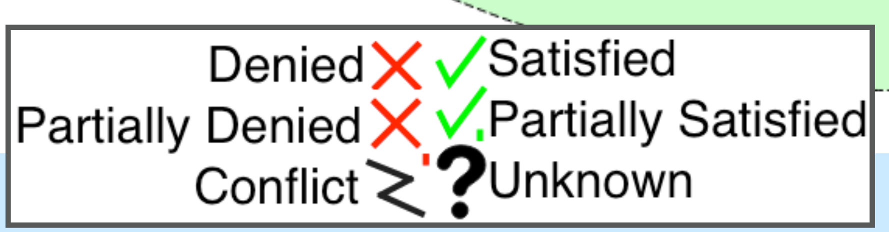
Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?

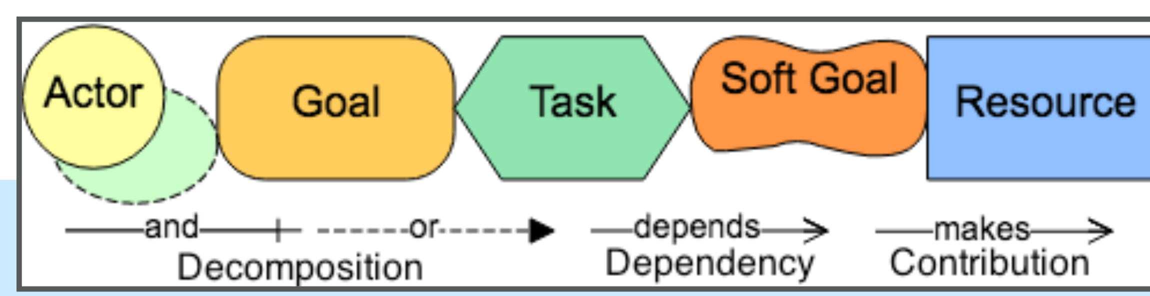


Waste Management Example

Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?







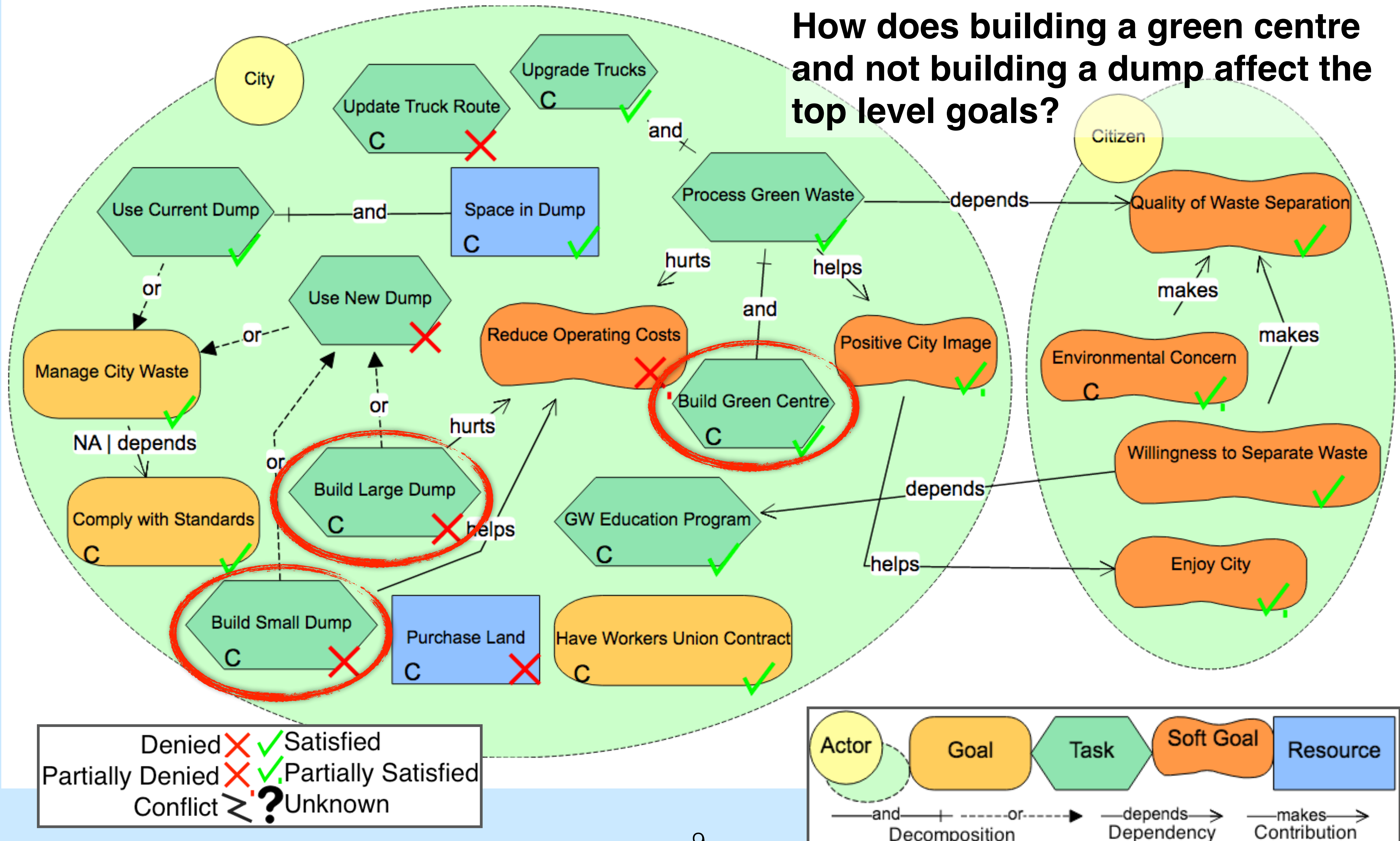
Waste Management Example

Question: Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?

Answer: Yes, by build a green centre and a small dump.

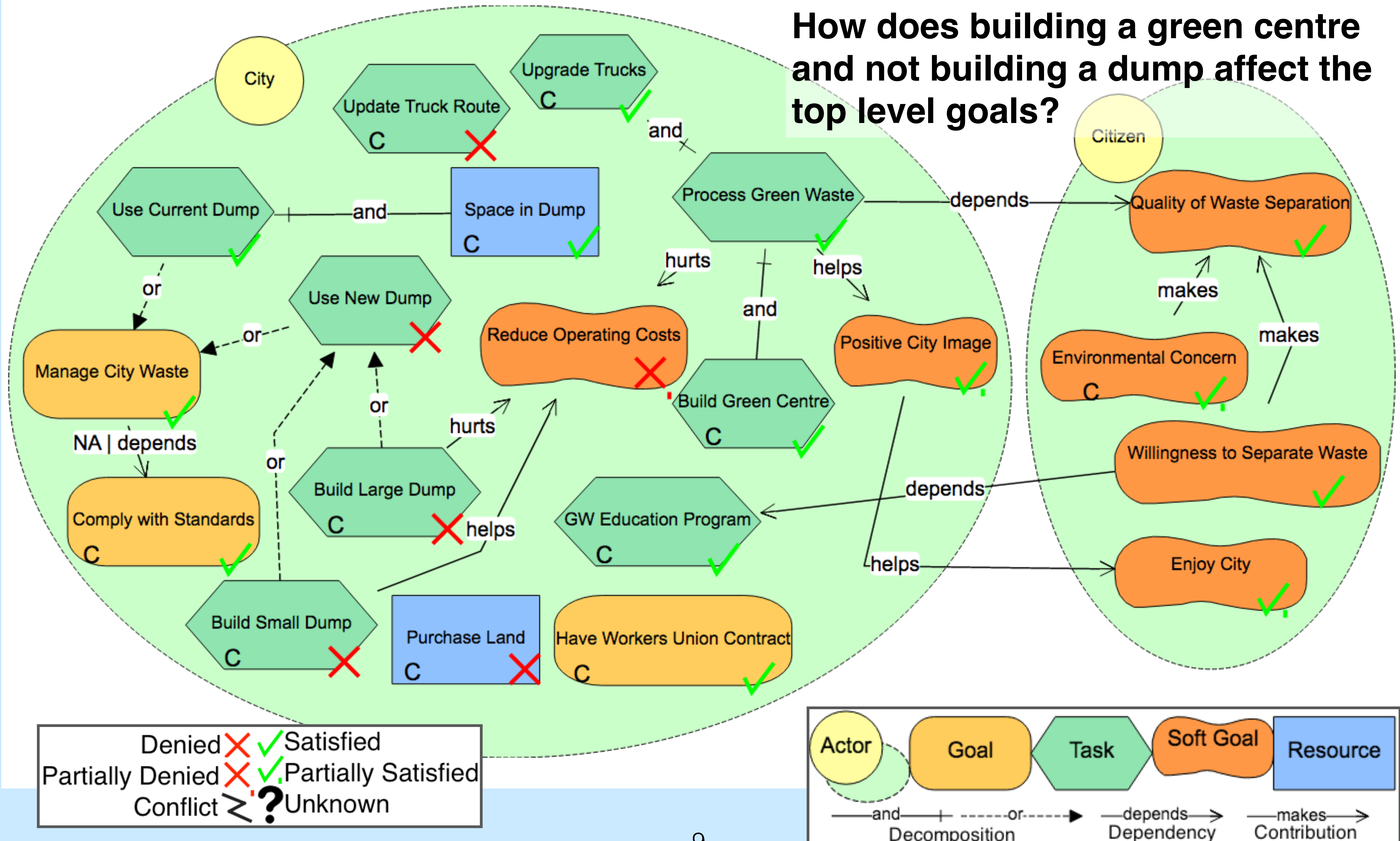
Waste Management Example

How does building a green centre and not building a dump affect the top level goals?



Waste Management Example

How does building a green centre and not building a dump affect the top level goals?



Waste Management Example

Question: How does building a green centre and not building a dump affect the top level goals?

Answer: It satisfies (or partially satisfies) the top goals, except Reduce Operating Costs.

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

See paper for additional questions…

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

See paper for additional questions…

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

?

See paper for additional questions…

Contributions

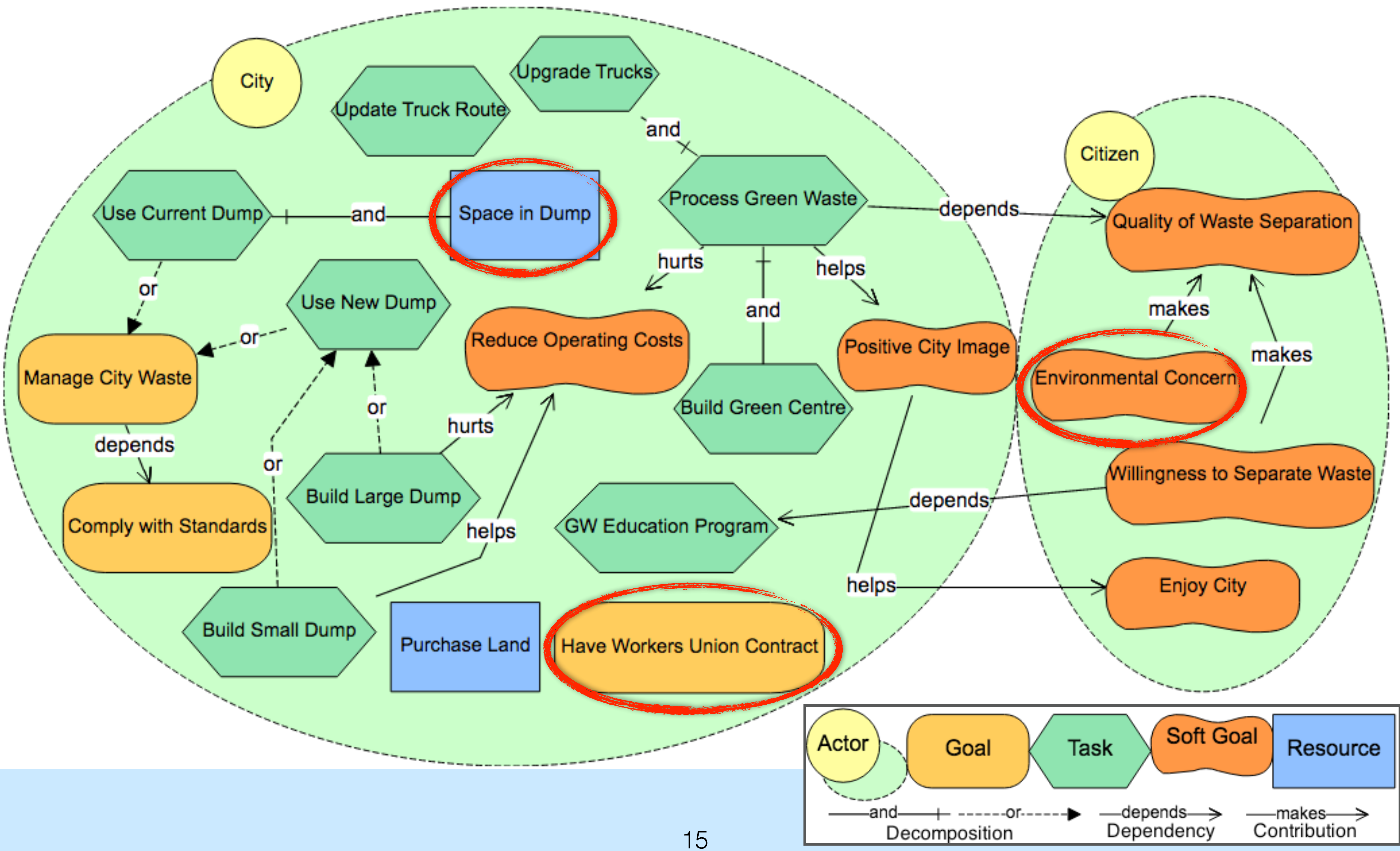
- Understand the impacts of dynamically changing intentions on decision making

- Enrich goal models intentions with dynamically changing evaluation

Outline

- Motivating Example - City Waste Management
- **Modeling Dynamic Intentions**
- Analysis Techniques with Dynamic Intentions
    - Simulation
    - CSP and CSP with Constraints
- Tooling and Validation
- Conclusion and Future Directions

Modeling Dynamic Intentions



Modeling Dynamic Intentions

Stochastic (R)

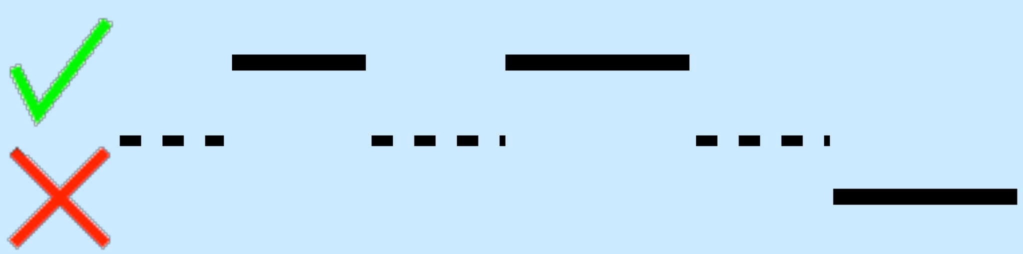
Patterns:

Examples:

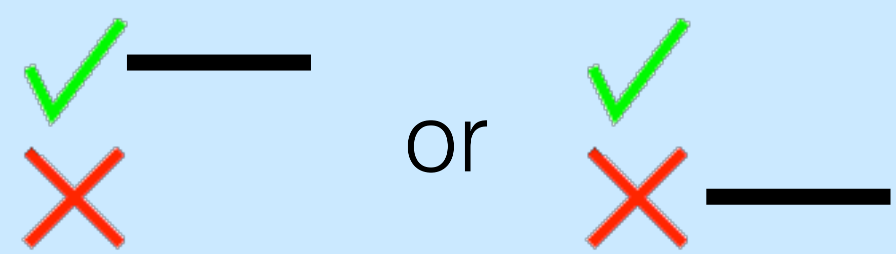
Modeling Dynamic Intentions

Elementary Functions

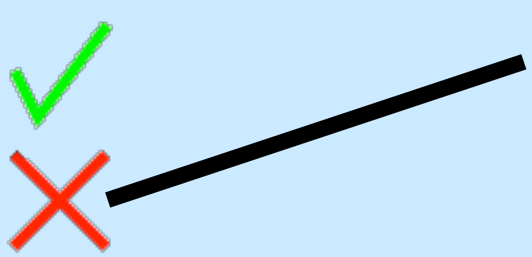
Stochastic (R):



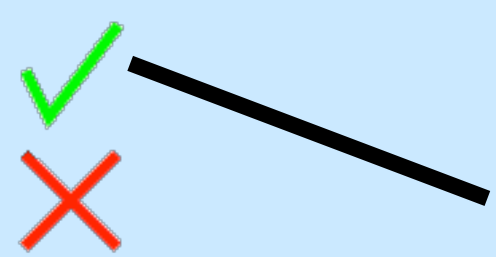
Constant (C):



Increase (I):



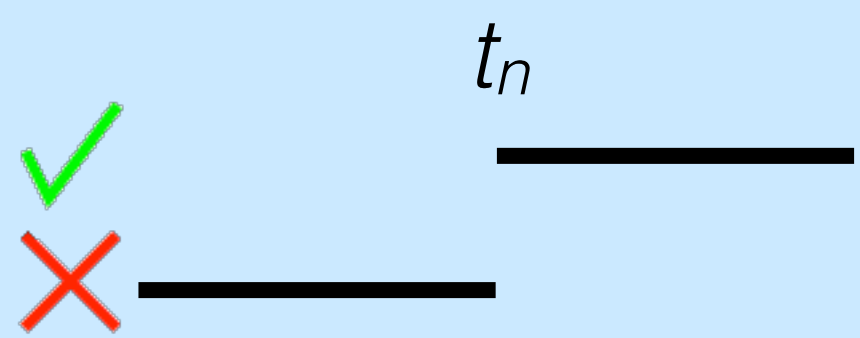
Decrease (D):



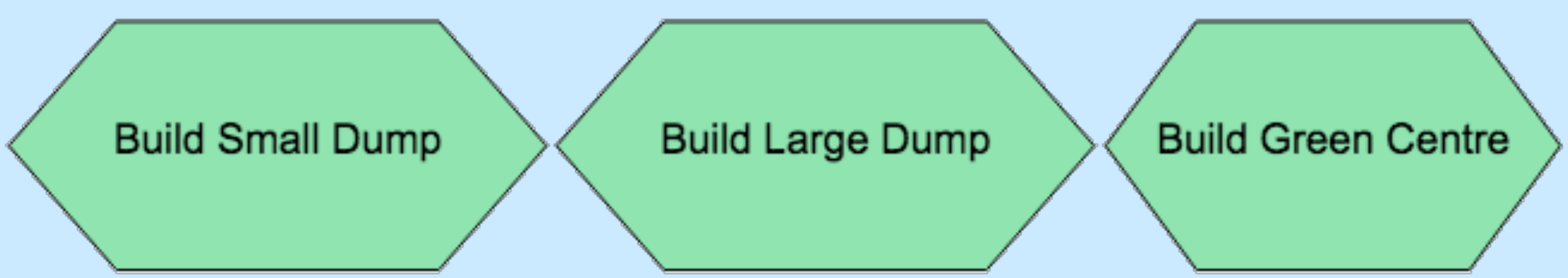
Modeling Dynamic Intentions

Denied-Satisfied (DS)

Patterns:



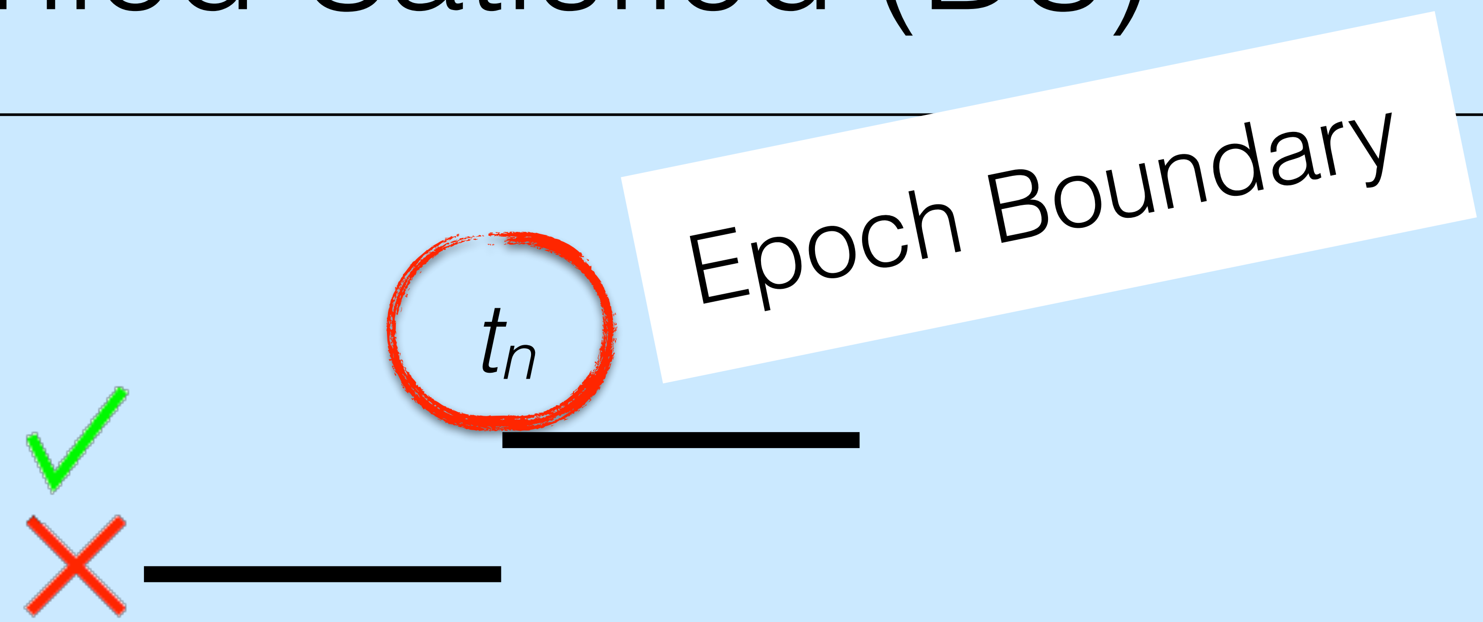
Examples:



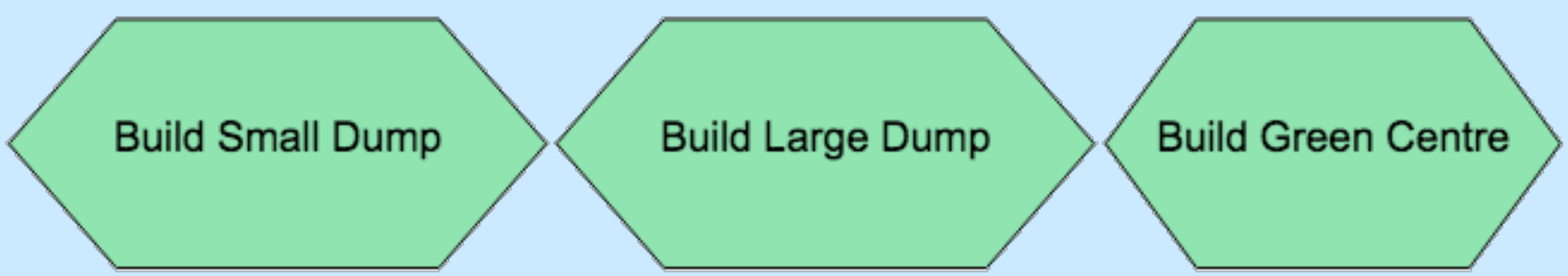
Modeling Dynamic Intentions

Denied-Satisfied (DS)

Patterns:



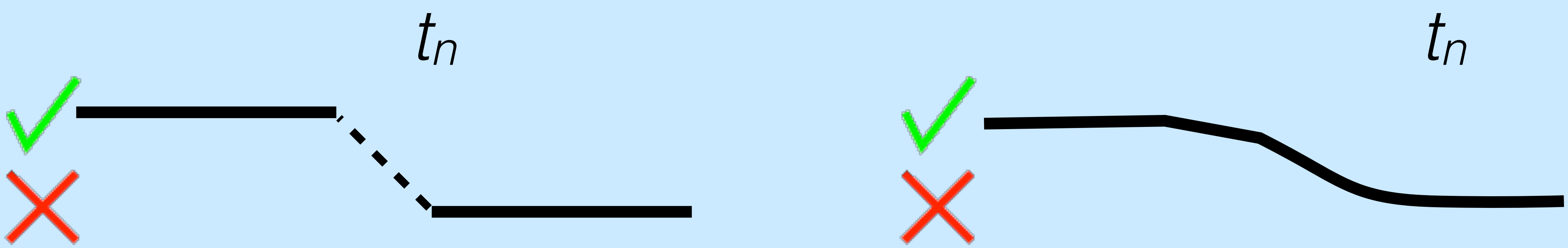
Examples:



Modeling Dynamic Intentions

Monotonic Negative (MN)

Patterns:



Examples:

Common Compound Functions

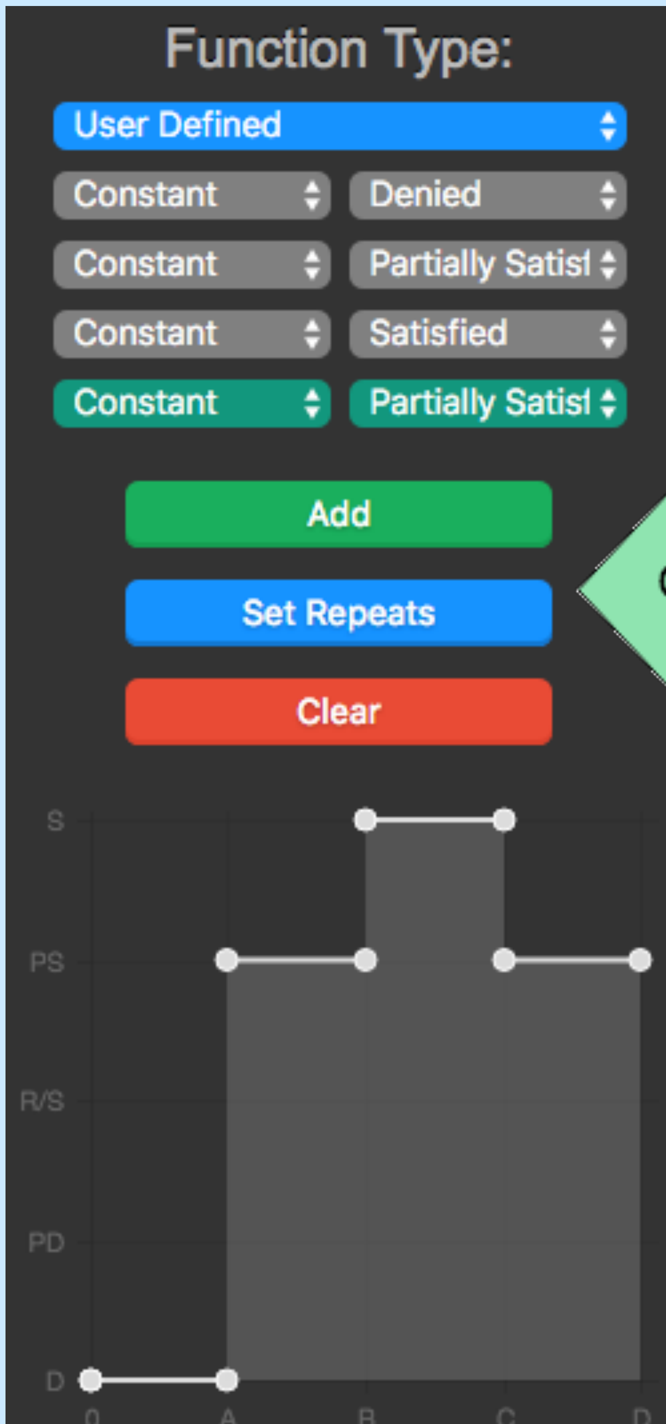
Denied-Satisfied (DS)	the satisfaction evaluation remains *Denied* until t_i and then remains *Satisfied*
Monotonic Negative (MN)	changes in satisfaction evaluation become "less true" to a *maxValue* at t_i and then remains constant at *constantValue*

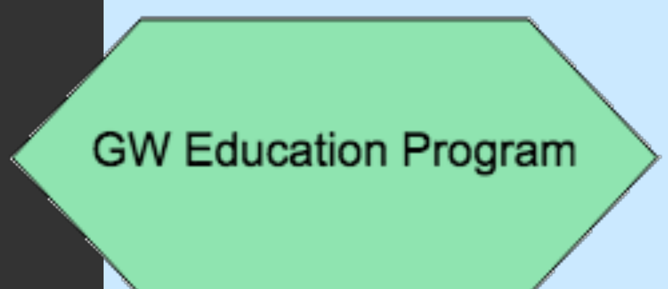
Common Compound Functions

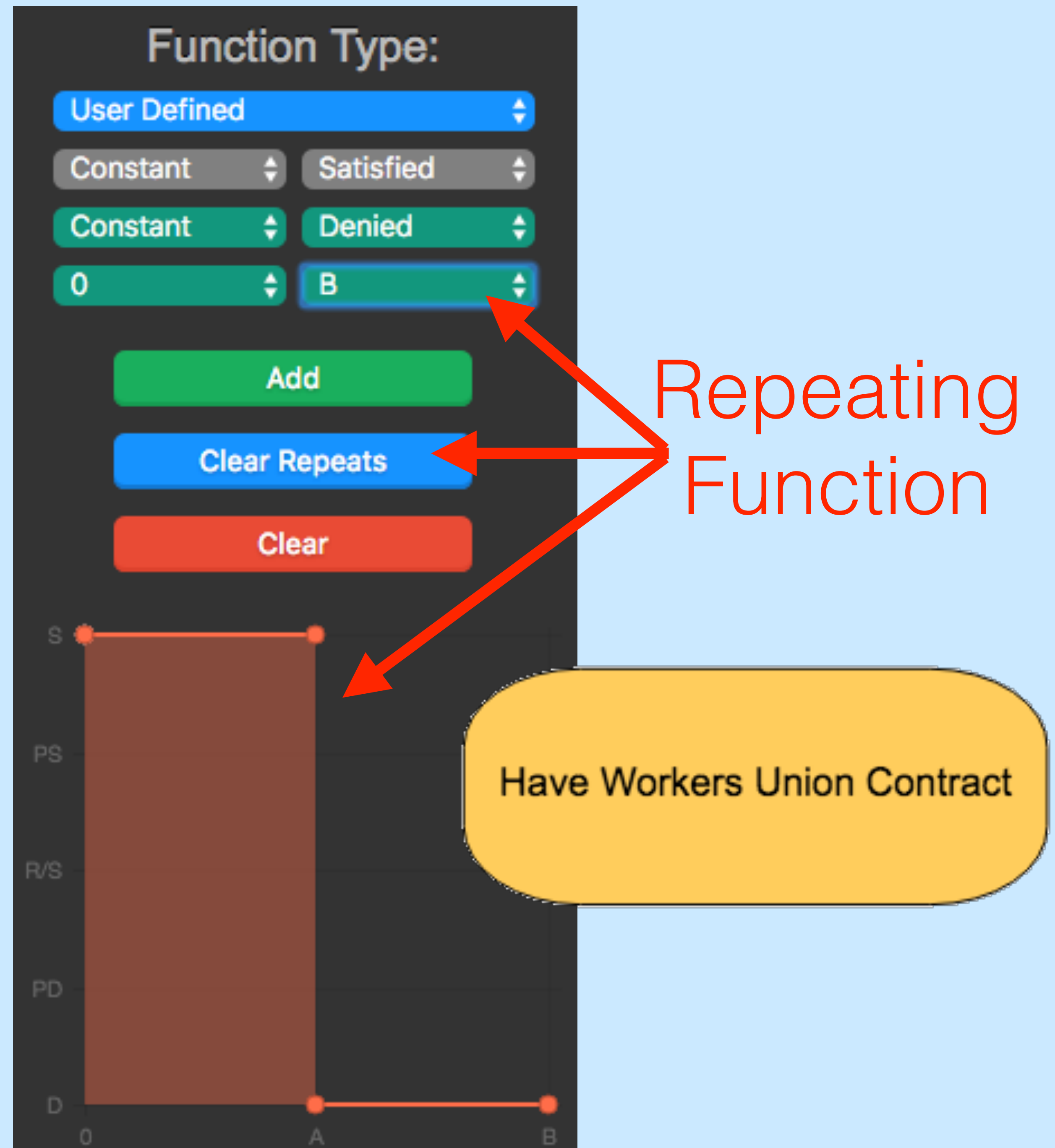
Satisfied-Denied (SD)	the satisfaction evaluation remains *Satisfied* until t_i and then remains *Denied*
Denied-Satisfied (DS)	the satisfaction evaluation remains *Denied* until t_i and then remains *Satisfied*
Stochastic-Constant (RC)	changes in satisfaction evaluation are stochastic or random until t_i and then remains constant at *constantValue*
Constant-Stochastic (CR)	the satisfaction evaluation remains constant at *constantValue* until t_i and then changes in evaluation are stochastic or random
Monotonic Positive (MP)	changes in satisfaction evaluation become "more true" to a *maxValue* at t_i and then remains constant at *constantValue*
Monotonic Negative (MN)	changes in satisfaction evaluation become "less true" to a *maxValue* at t_i and then remains constant at *constantValue*

Modeling Dynamic Intentions

User Defined (UD)







Outline

- Motivating Example - City Waste Management
- Modeling Dynamic Intentions
- **Analysis Techniques with Dynamic Intentions**
    - Simulation
    - CSP and CSP with Constraints
- Tooling and Validation
- Conclusion and Future Directions

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

Strategies

(Strategy 1) create a **random path** given initial states in the model

(Strategy 2) create a path given **desired properties** of the **intermediate state** (with optional properties over the initial or final state)

(Strategy 3) create a path which is **different than the previously seen path** over the same constraints

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

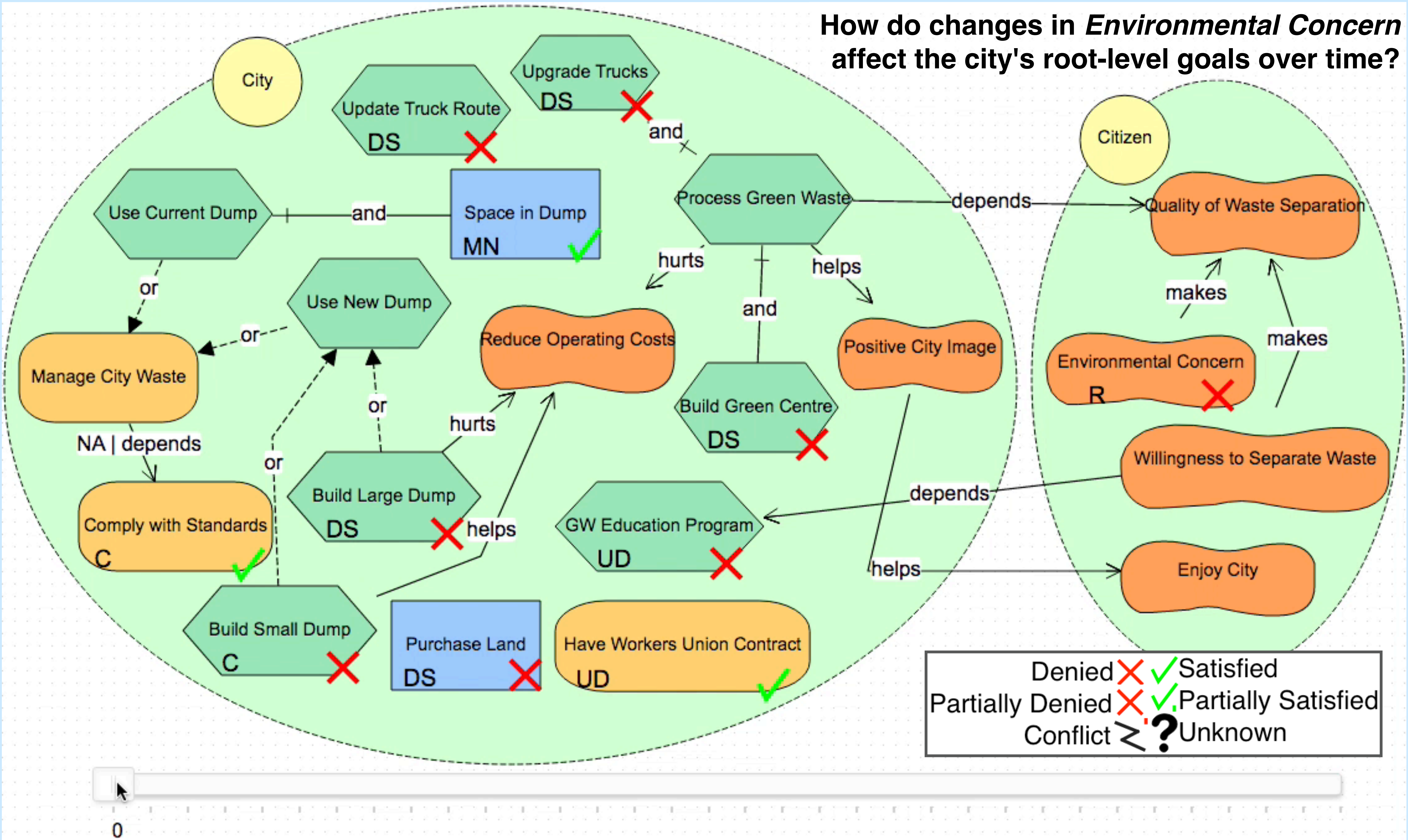
Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenari… *City Waste* ever… denied in the fu…
5. Does the order o… developments (*Process Green Waste* and *Use New Dump*) matter?

Strategy 1: create a random path given **initial states** in the model
- Leaf Simulation -

Leaf Simulation (Initial States)

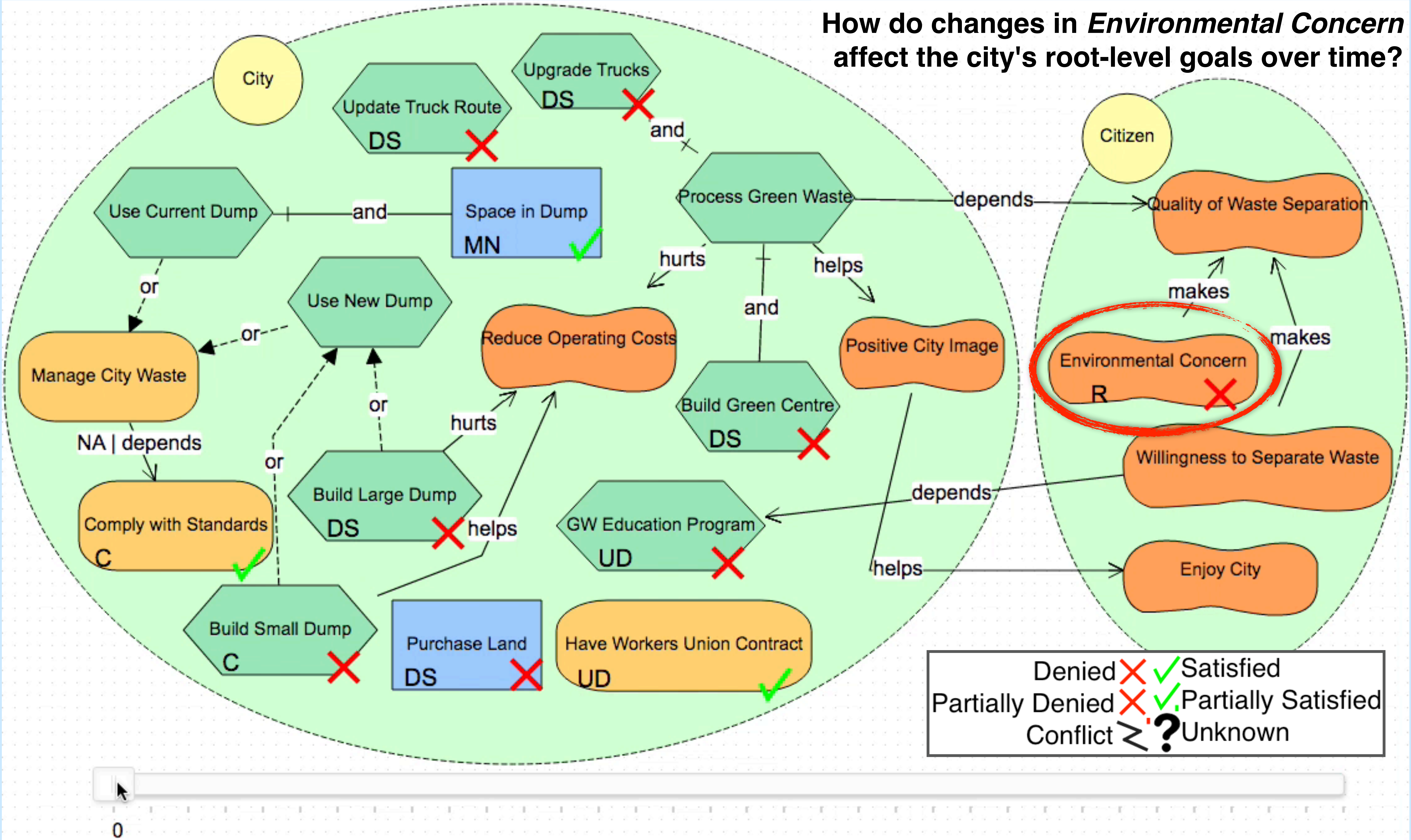
How do changes in *Environmental Concern* affect the city's root-level goals over time?



R:Stochastic, C:Constant, UD:User Defined

DS:Denied-Satisfied, MN:Monotonic Negative

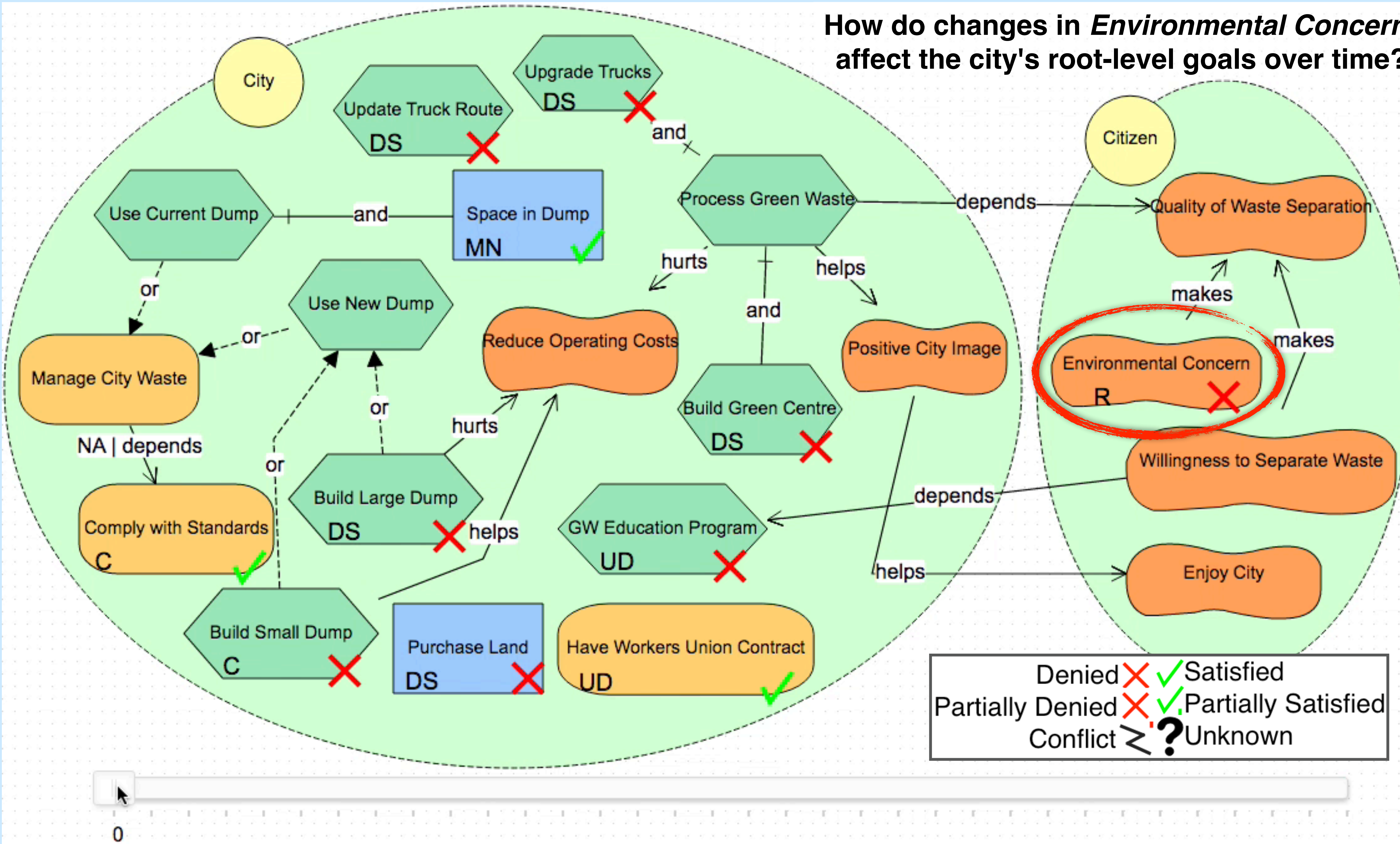
Leaf Simulation (Initial States)



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

Leaf Simulation (Initial States)

How do changes in *Environmental Concern* affect the city's root-level goals over time?



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

Waste Management Example

Question: How do changes in Environmental Concern affect the city's root-level goals over time?

Answer: Affects Reduced Operating Cost and Enjoy City. Having a GW Education Program mitigates the effect of denied environmental concern.

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

Waste Management Example

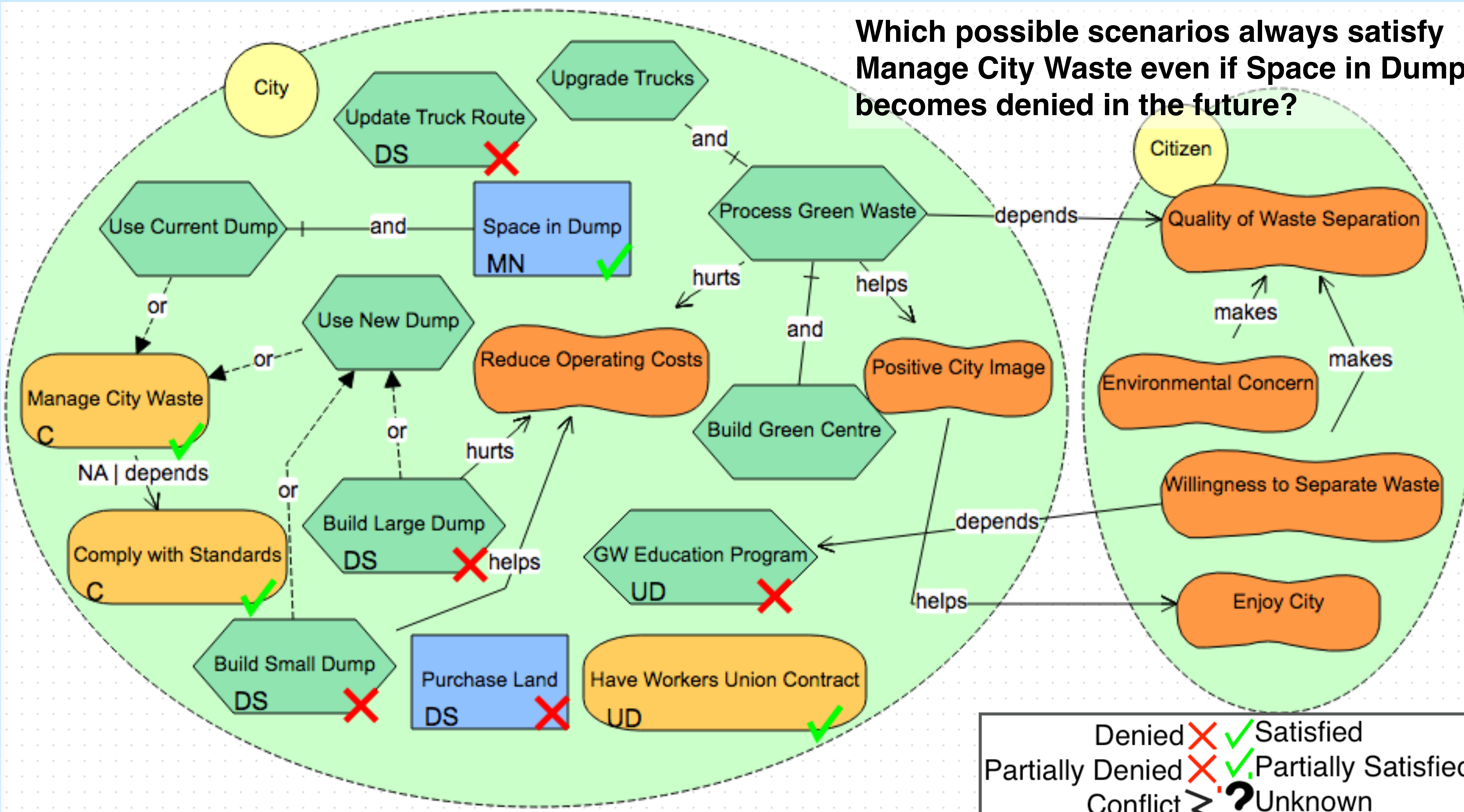
1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. H…
3. … affect … level goals over time?

Strategy 2: create a path given desired properties of the **intermediate state** (with optional properties over the initial or **final state**)

- CSP Analysis -

4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

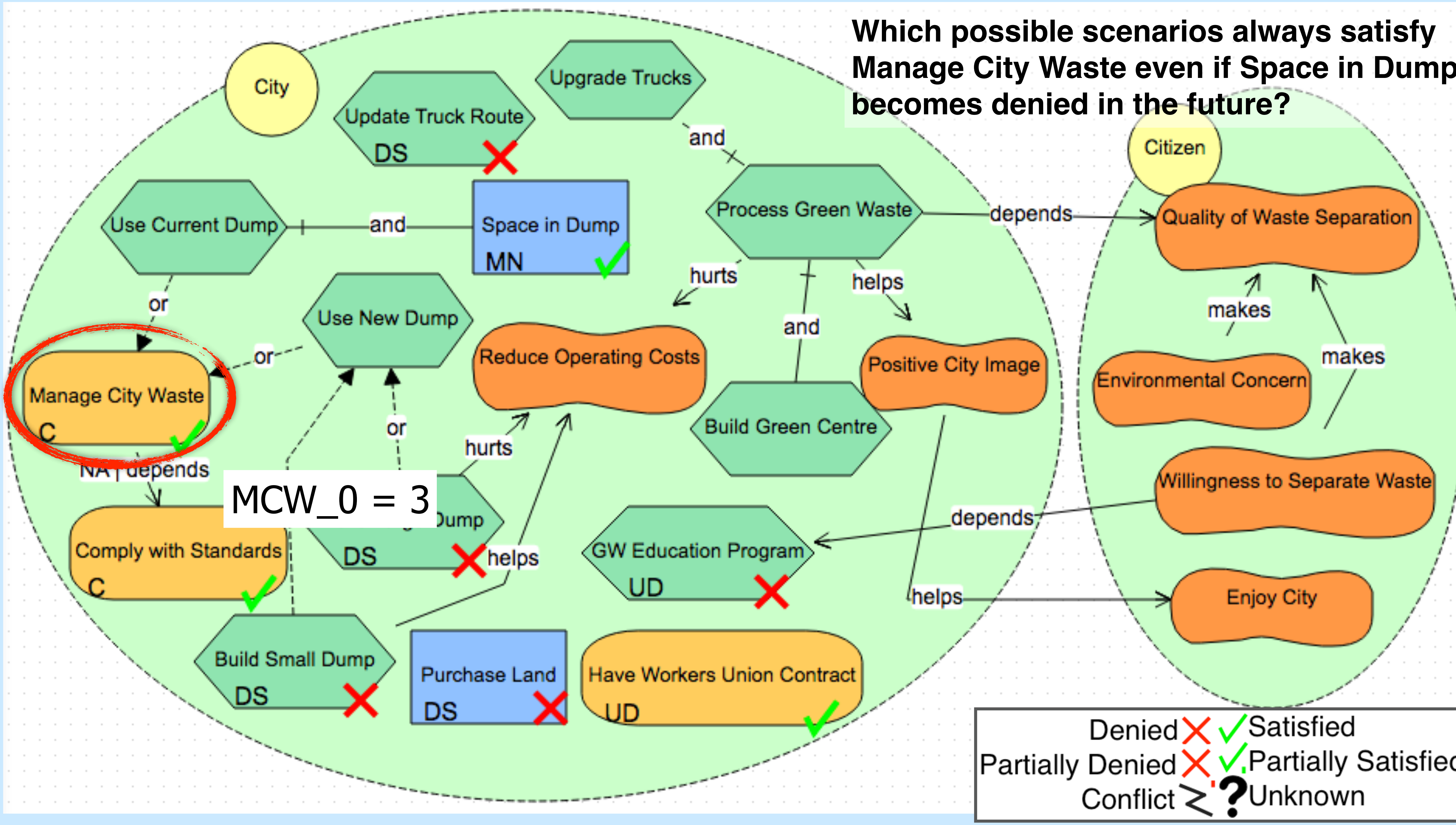
CSP Analysis (Intermediate/Final)



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

CSP Analysis (Intermediate/Final)

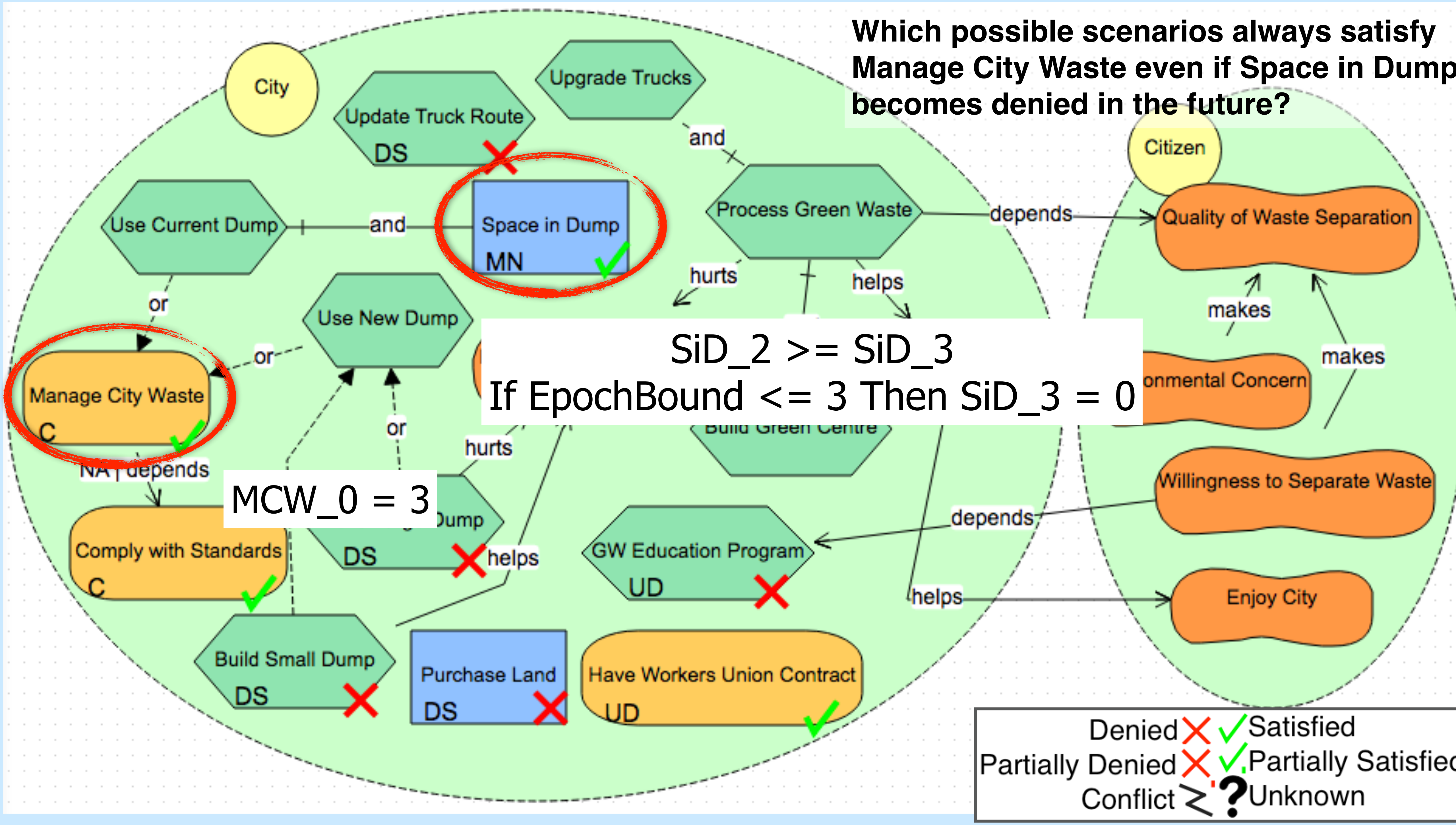
Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

CSP Analysis (Intermediate/Final)

Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?



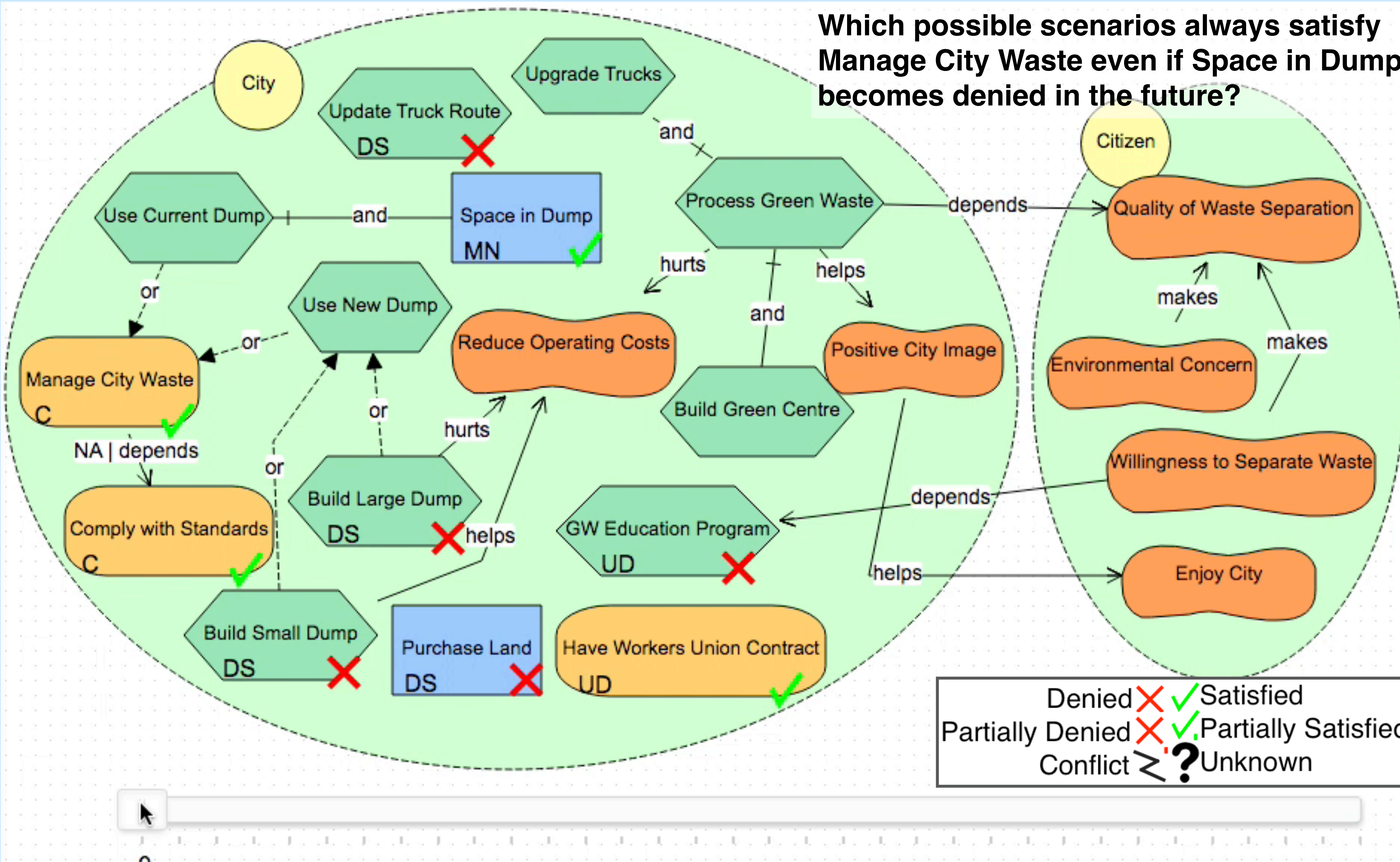
SiD_2 >= SiD_3

If EpochBound <= 3 Then SiD_3 = 0

MCW_0 = 3

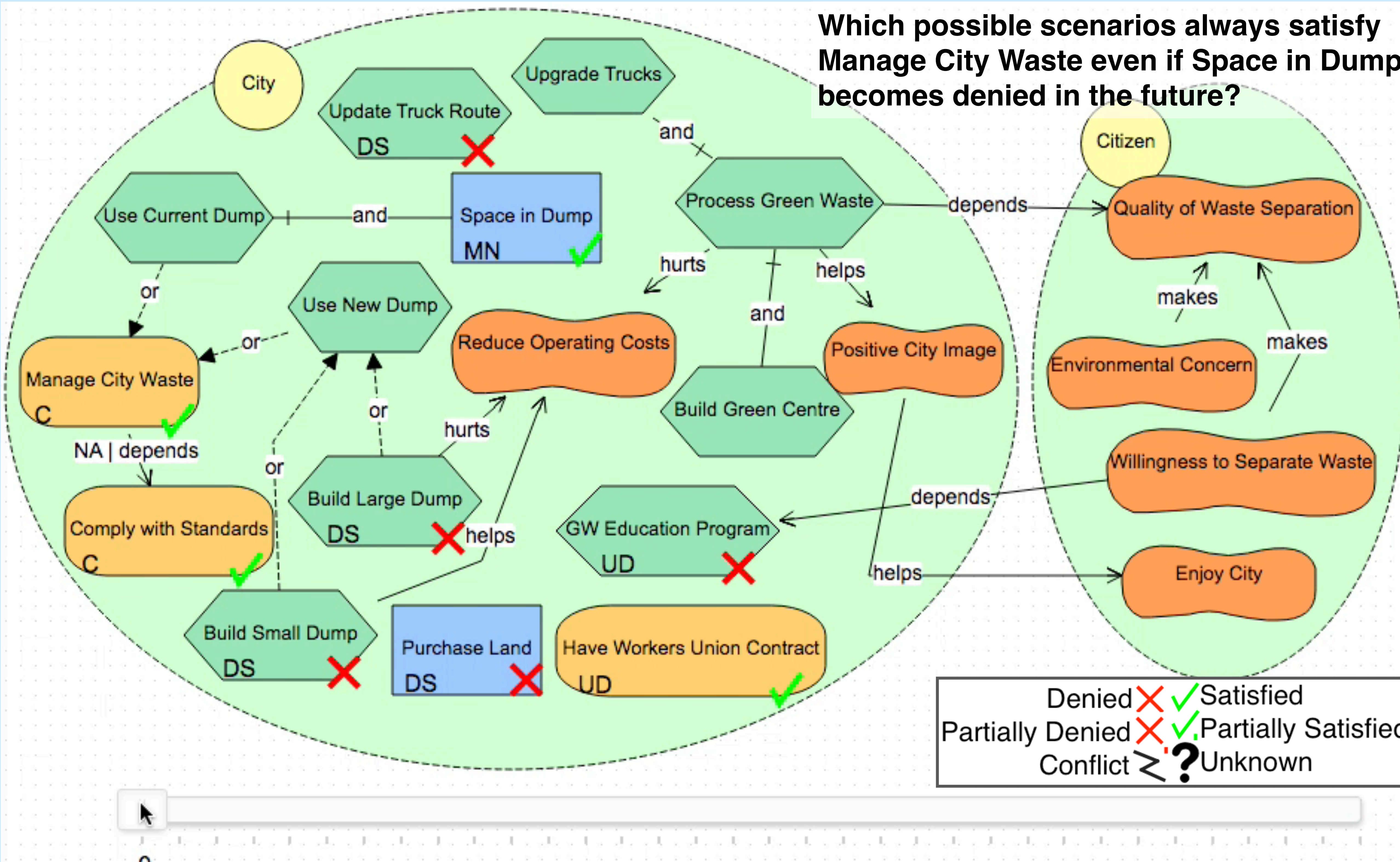
R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

CSP Analysis (Intermediate/Final)



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

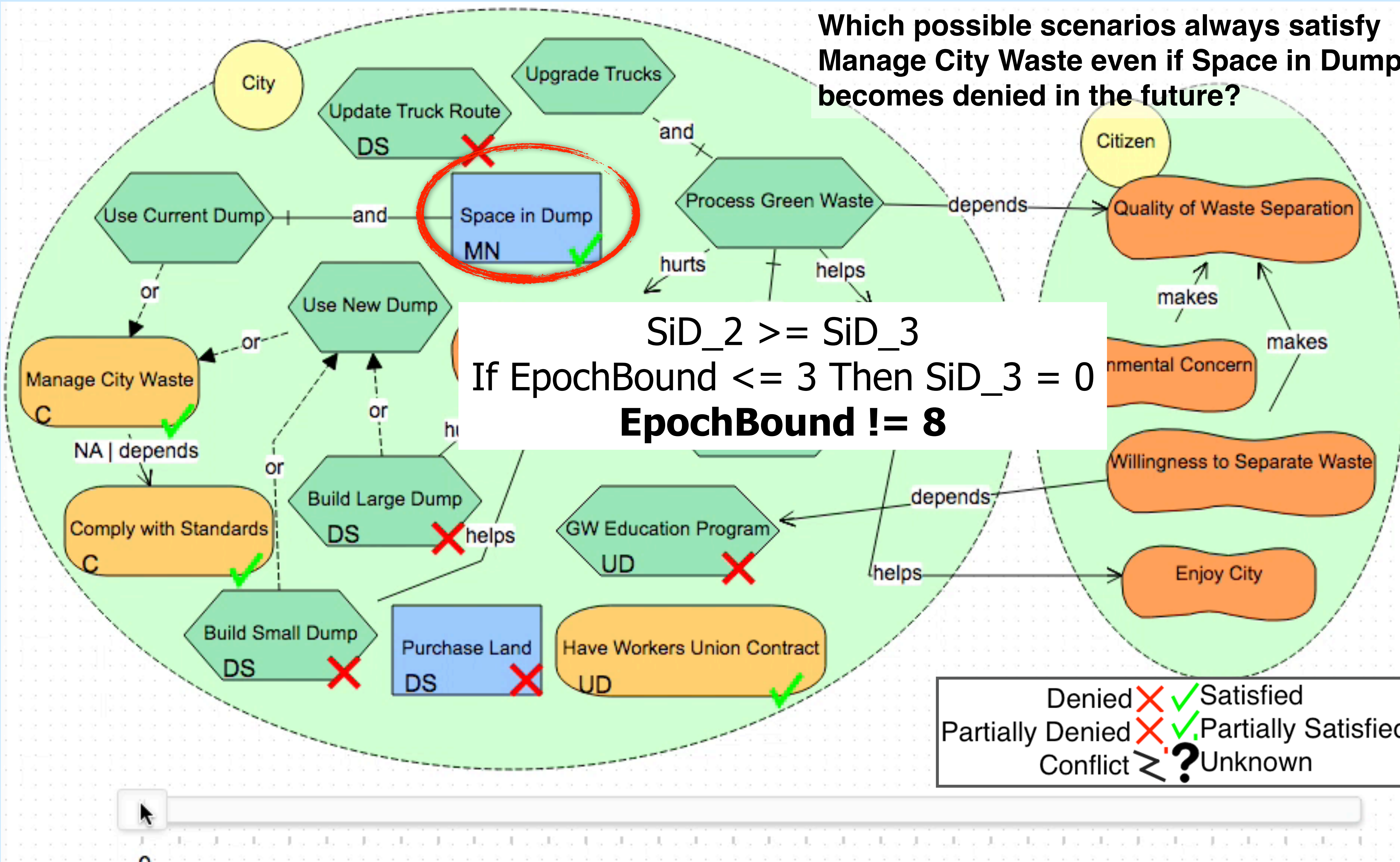
CSP Analysis (Intermediate/Final)



R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

CSP Analysis (Intermediate/Final)

Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?



SiD_2 >= SiD_3

If EpochBound <= 3 Then SiD_3 = 0

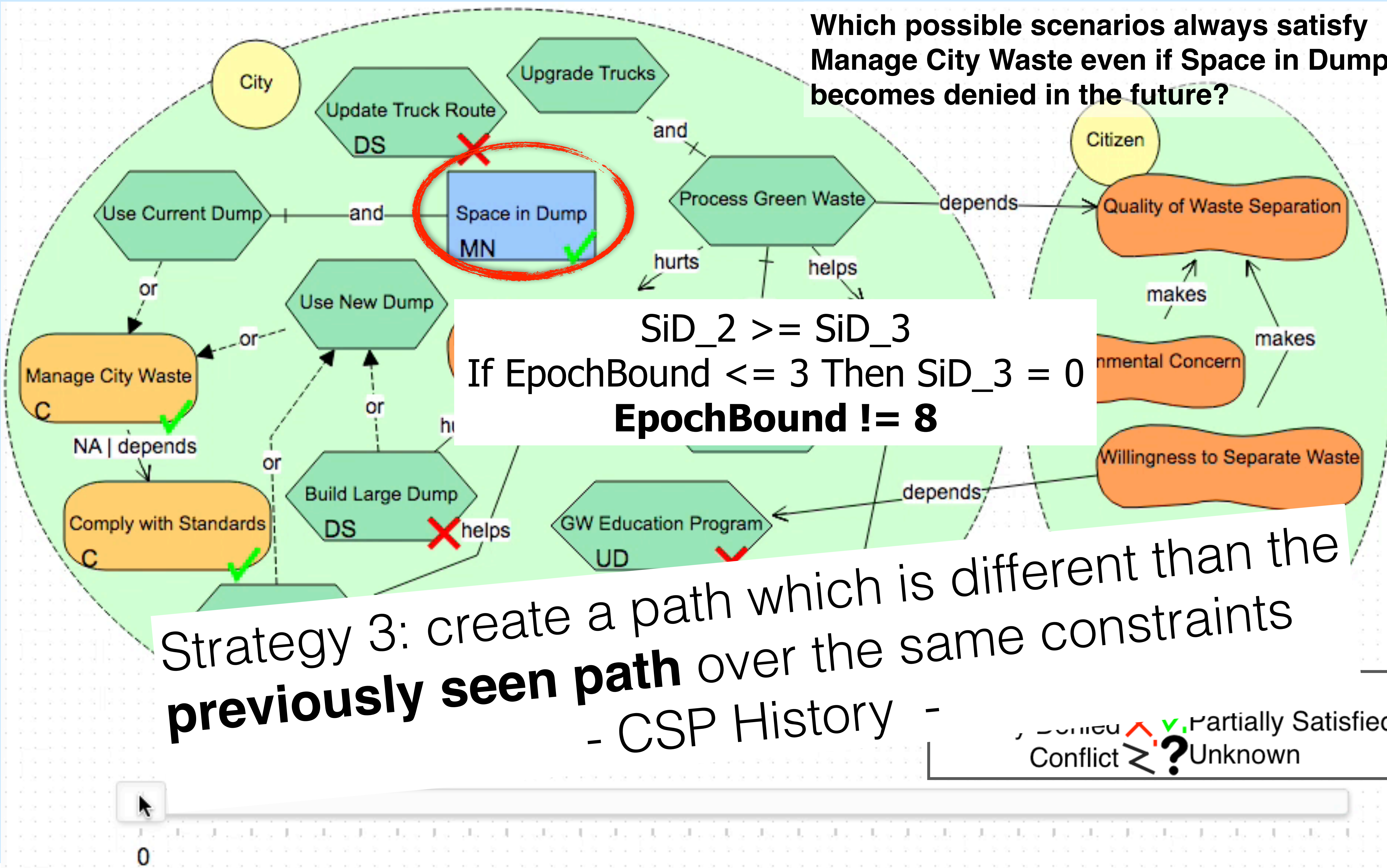
EpochBound != 8

R:Stochastic, C:Constant, UD:User Defined

DS:Denied-Satisfied, MN:Monotonic Negative

CSP Analysis (Intermediate/Final)

Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?



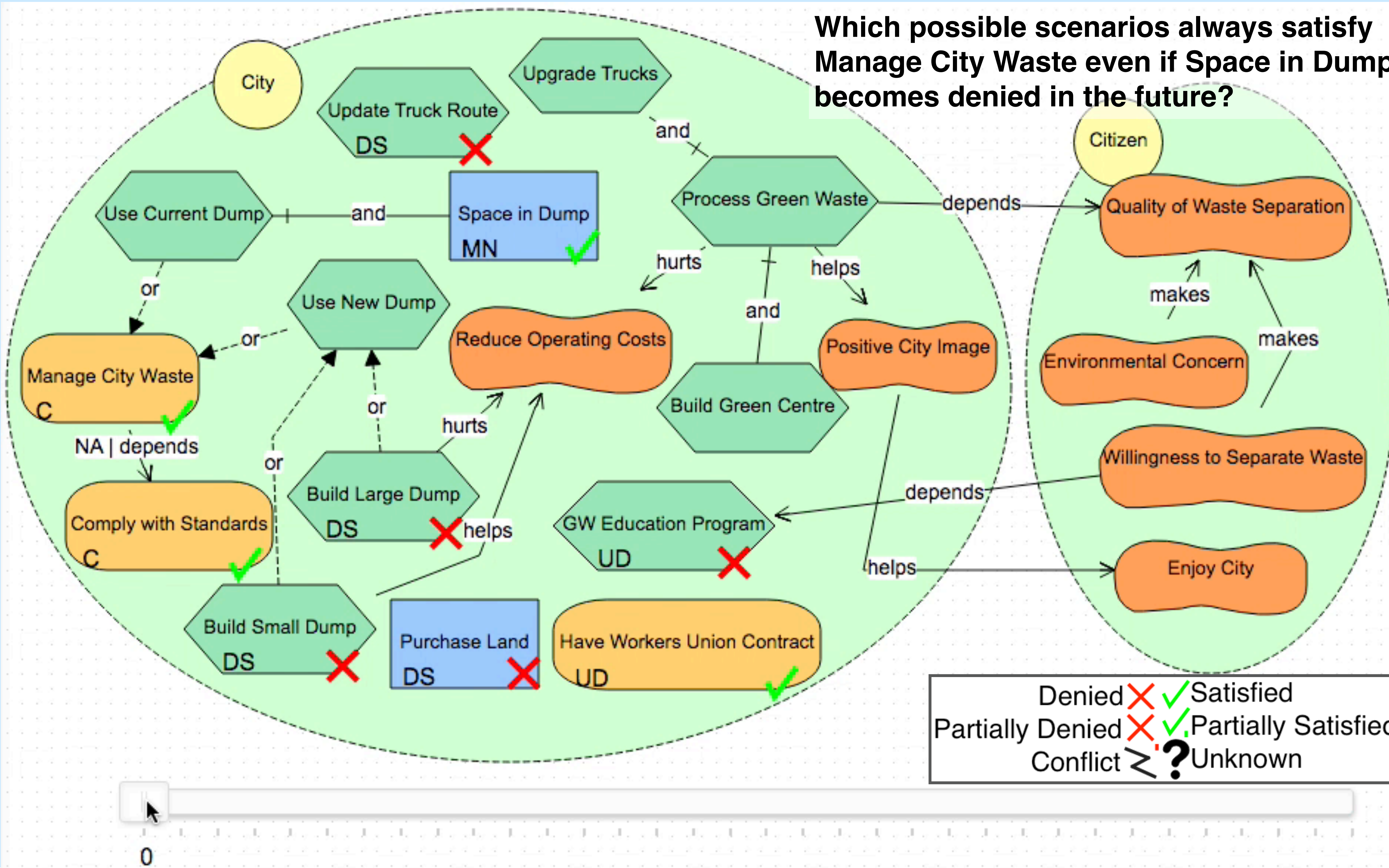
SiD_2 >= SiD_3
If EpochBound <= 3 Then SiD_3 = 0
EpochBound != 8

Strategy 3: create a path which is different than the **previously seen path** over the same constraints
- CSP History -

R:Stochastic, C:Constant, UD:User Defined DS:Denied-Satisfied, MN:Monotonic Negative

CSP History (Previous Path)

Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?

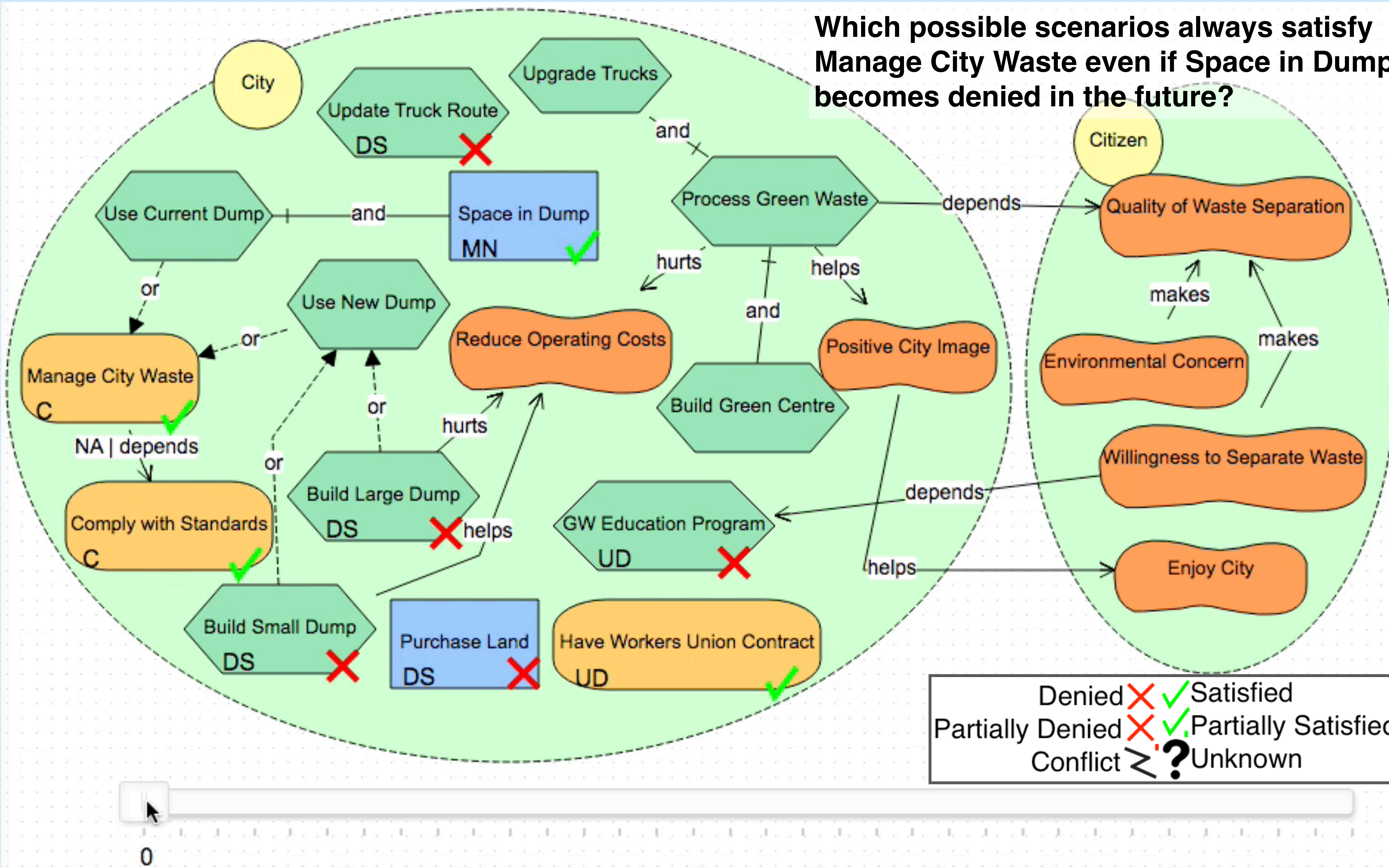


R:Stochastic, C:Constant, UD:User Defined

DS:Denied-Satisfied, MN:Monotonic Negative

CSP History (Previous Path)

Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?



R:Stochastic, C:Constant, UD:User Defined

DS:Denied-Satisfied, MN:Monotonic Negative

Waste Management Example

Question: Which possible scenarios always satisfy Manage City Waste even if Space in Dump becomes denied in the future?

Answer: Build Large Dump must be satisfied prior to Space in Dump becoming denied.

Note: Build Small Dump also suffices (paths not shown).

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environm*…
4. … future?

Strategy 2: create a path given desired properties of the **intermediate state** (with optional properties over the initial or **final state**)
- CSP Analysis (with Queries) -

5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

CSP Analysis (with Queries)

Does the order of these developments (Process Green Waste and Use New Dump) matter?

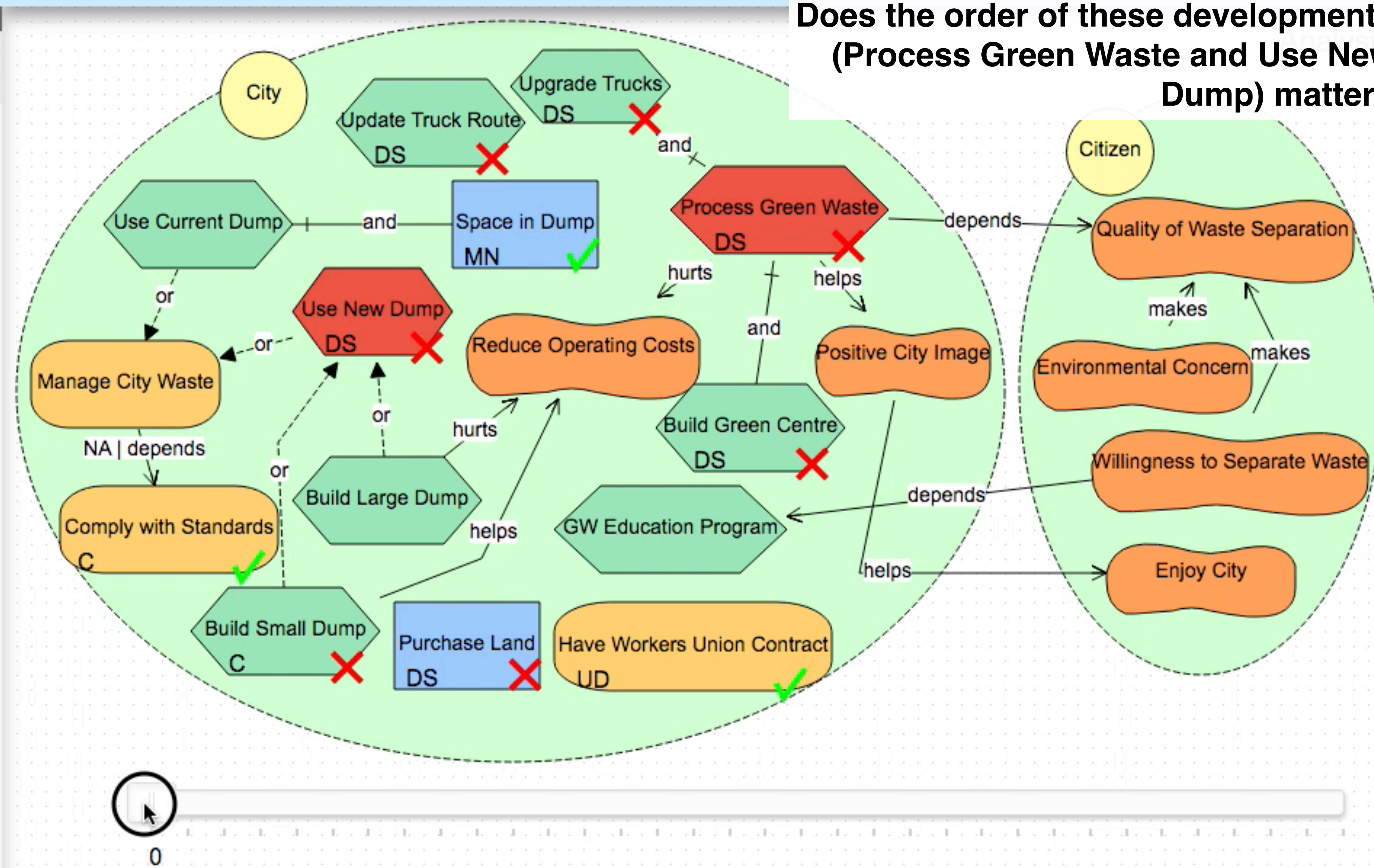
History Log

Step 1: CSP A<B

Step 2: CSP B<A

Step 3: CSP A=B

A: Process Green Waste
B: Use New Dump



R:Stochastic, C:Constant, UD:User Defined
DS:Denied-Satisfied, MN:Monotonic Negative

Denied ✗ ✓ Satisfied
Partially Denied ✗ ✓ Partially Satisfied
Conflict ? Unknown

CSP Analysis (with Queries)

Does the order of these developments (Process Green Waste and Use New Dump) matter?

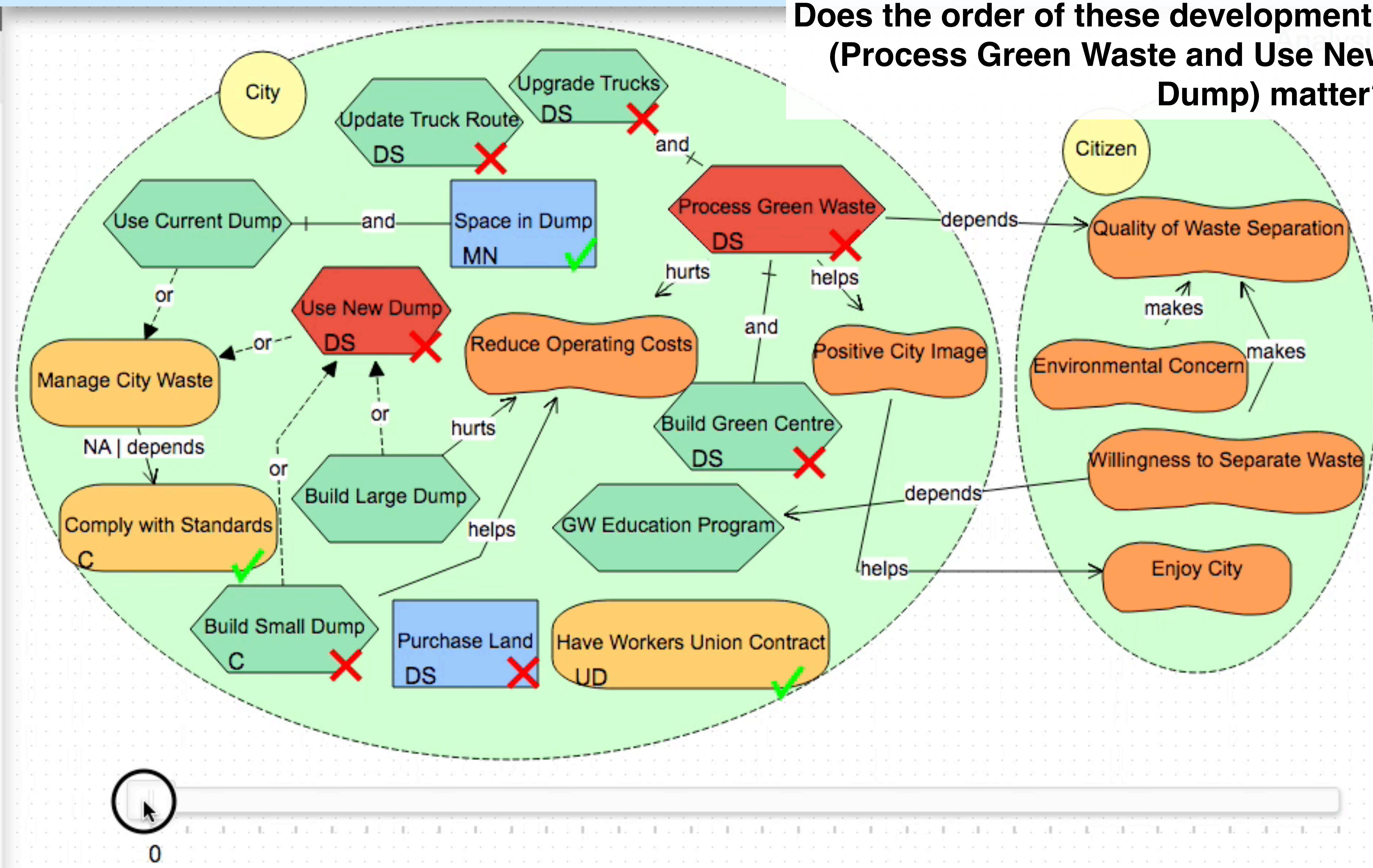
History Log

Step 1: CSP A<B

Step 2: CSP B<A

Step 3: CSP A=B

A: Process Green Waste
B: Use New Dump



R:Stochastic, C:Constant, UD:User Defined
DS:Denied-Satisfied, MN:Monotonic Negative

Denied | Satisfied
Partially Denied | Partially Satisfied
Conflict | Unknown

Waste Management Example

Question: Does the order of these developments (Process Green Waste and Use New Dump) matter?

Answer: No, given space in current dump.

Waste Management Example

1. Is it possible to satisfy Manage City Waste and partially satisfy Enjoy City? and how?
2. How does building a green centre and not building a dump affect the top level goals?
3. How do changes in *Environmental Concern* affect the city's root-level goals over time?
4. Which possible scenarios always satisfy *Manage City Waste* even if *Space in Dump* becomes denied in the future?
5. Does the order of these developments (*Process Green Waste* and *Use New Dump*) matter?

See paper for additional questions…

Waste Management Example

1. Build Green Centre and Build Small Dump is a possible scenario.
2. Building only Green Centre satisfies (or partially satisfies) the top goals, except Reduce Operating Costs.
3. Environmental Concern affects Reduced Operating Cost and Enjoy City over time. Having a GW Education Program mitigates the effect of denied Environmental Concern.
4. Build Large Dump (or Build Small Dump) must be satisfied prior to Space in Dump becoming denied.
5. Order of *Process Green Waste* and *Use New Dump doesn't matter*, given Space in Dump is not denied.

See paper for additional answers…

Waste Management Review

Goal: Evaluate waste management infrastructure

Intentions: Wants to be green and satisfy customer

Options: Build Green Centre
Build Landfill / Dump (large, small)

Solution (Standard): Build Green Centre

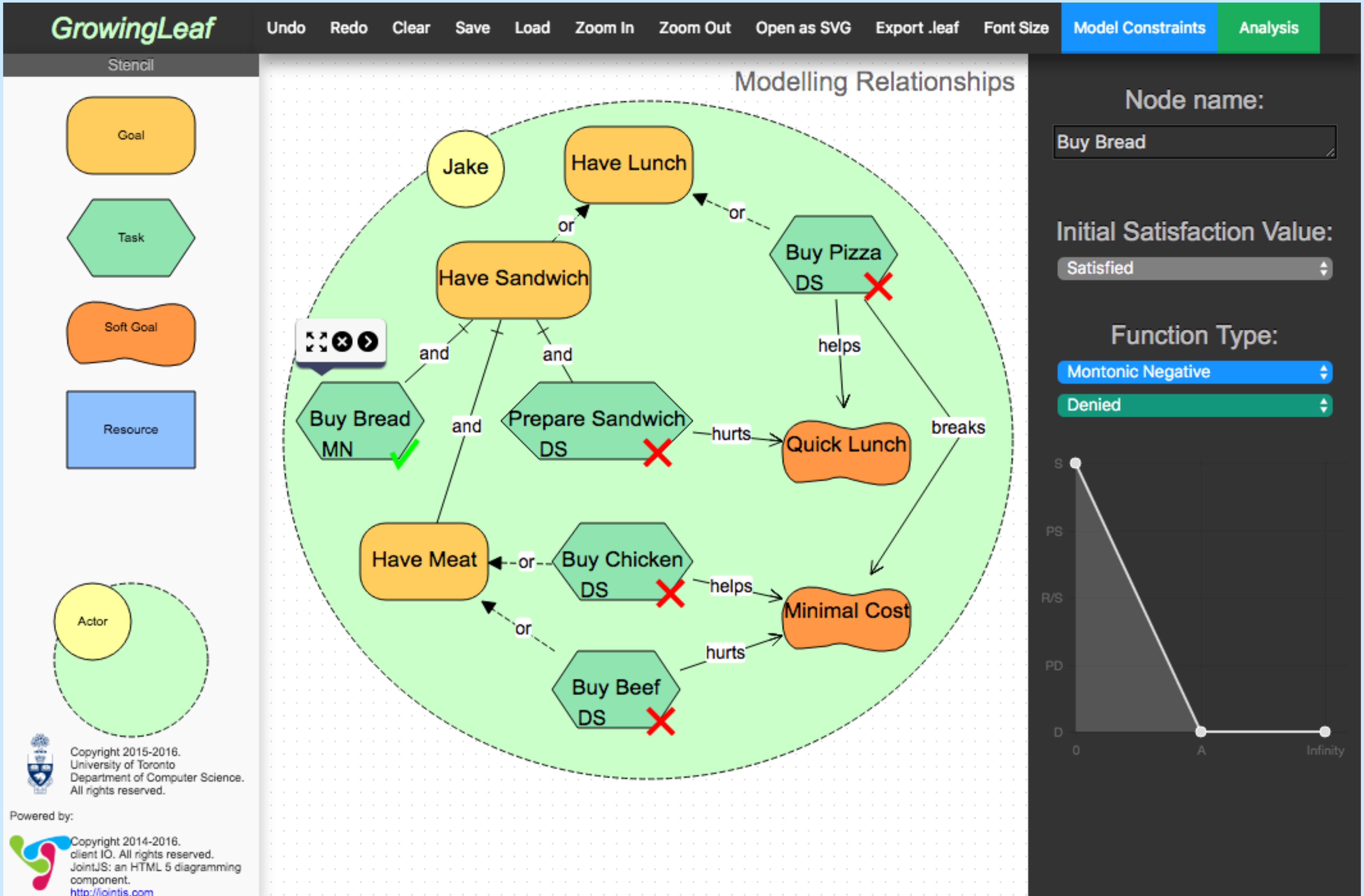
Solution (with Dynamics): Build Small Dump then
Build Green Centre

Outline

- Motivating Example - City Waste Management
- Modeling Dynamic Intentions
- Analysis Techniques with Dynamic Intentions
    - Simulation
    - CSP and CSP with Constraints
- **Tooling and Validation**
- Conclusion and Future Directions

Tooling: GrowingLeaf

http://www.cs.toronto.edu/~**amgrubb/growing-leaf**



Examples and Case Studies

- City transportation planning
- Network maintenance
- Software supply chains
- Technical debt
- Compliance
- Sustainability

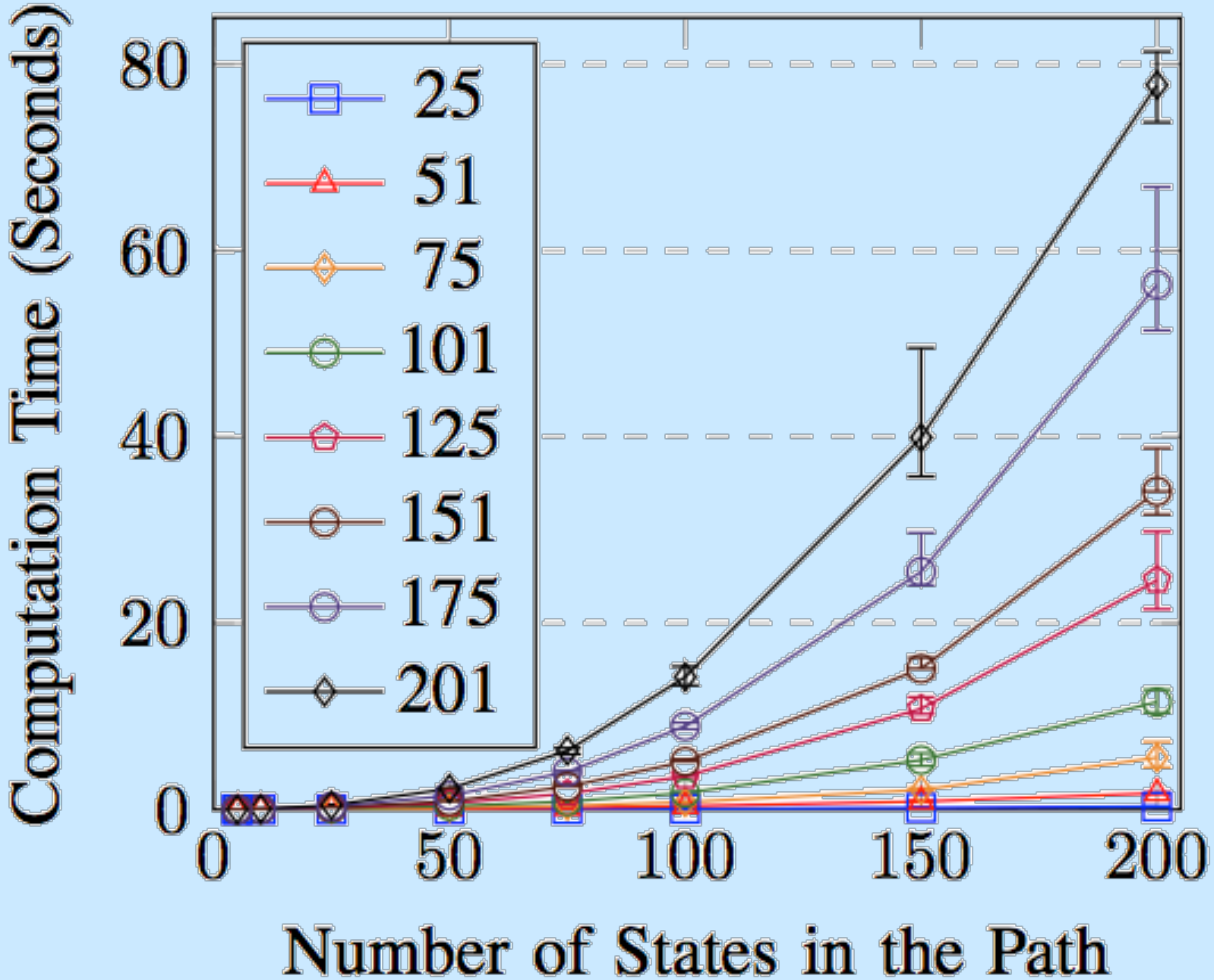
Further case studies are ongoing….

Scalability

- How does the length of the generated path affect the computation time in Strategy 1 and 2?
- How does the number of intentions in a model affect the computation time in Strategy 2?
- How does the number of previous paths used affect the computation time in Strategy 3?

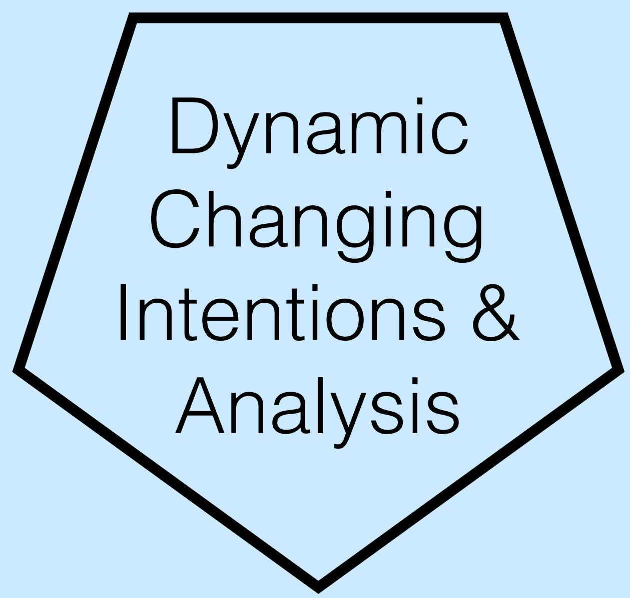
Details in the paper…

Scalability: Model Size

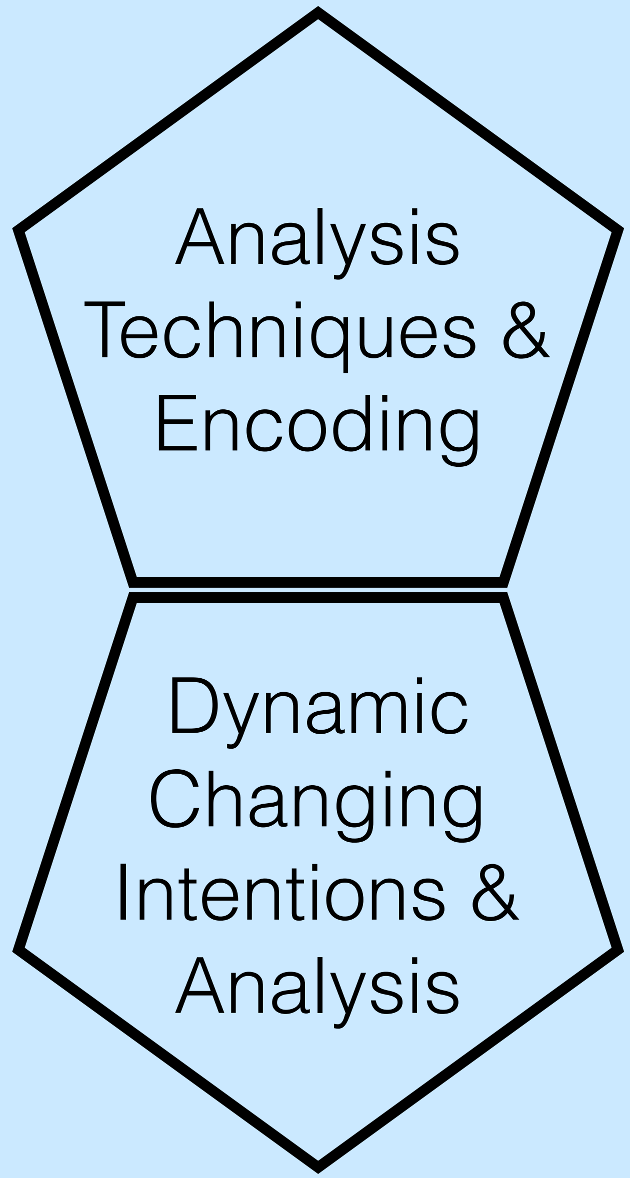


Results of changing the model size for CSP Analysis.

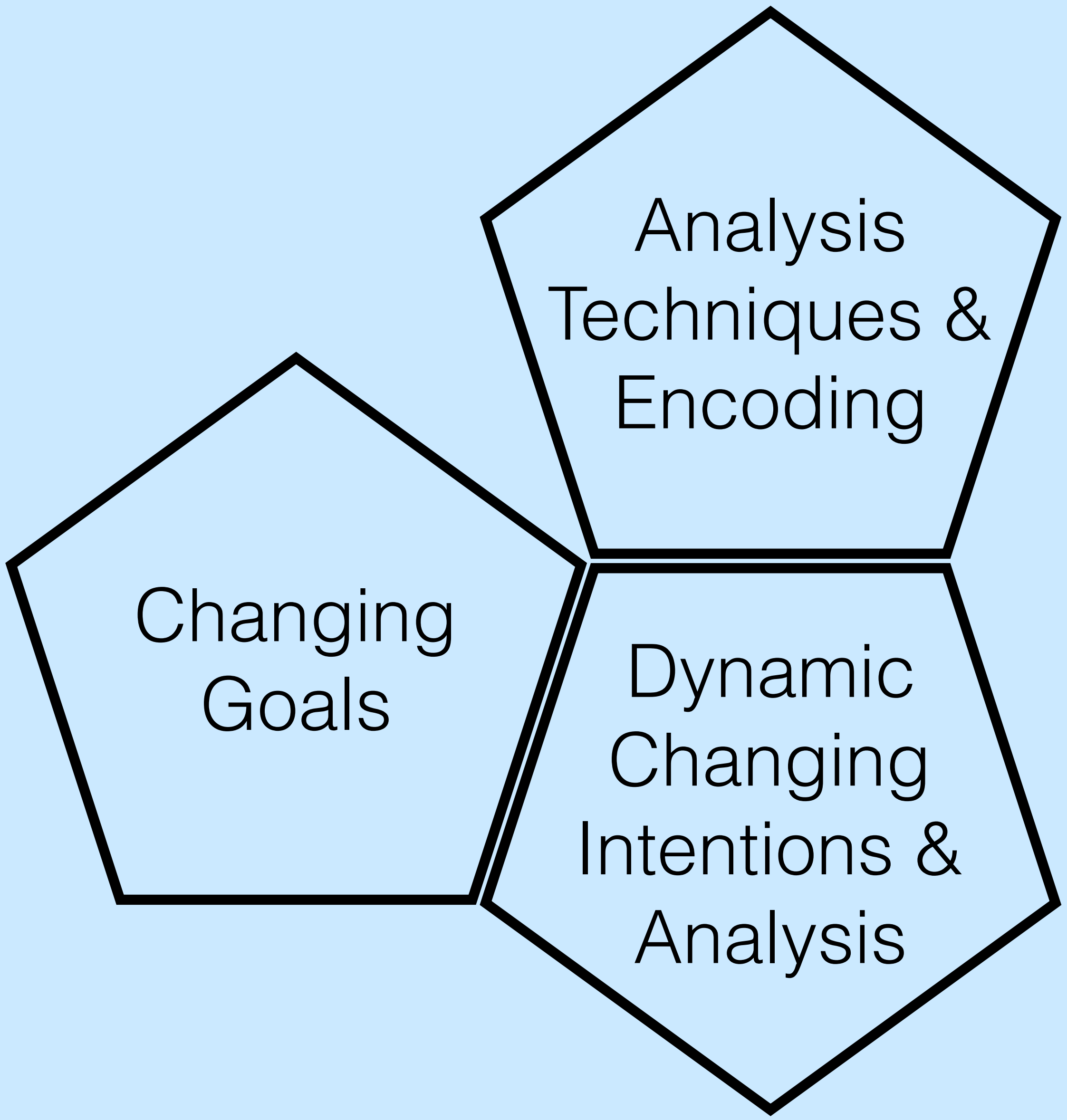
Related Work



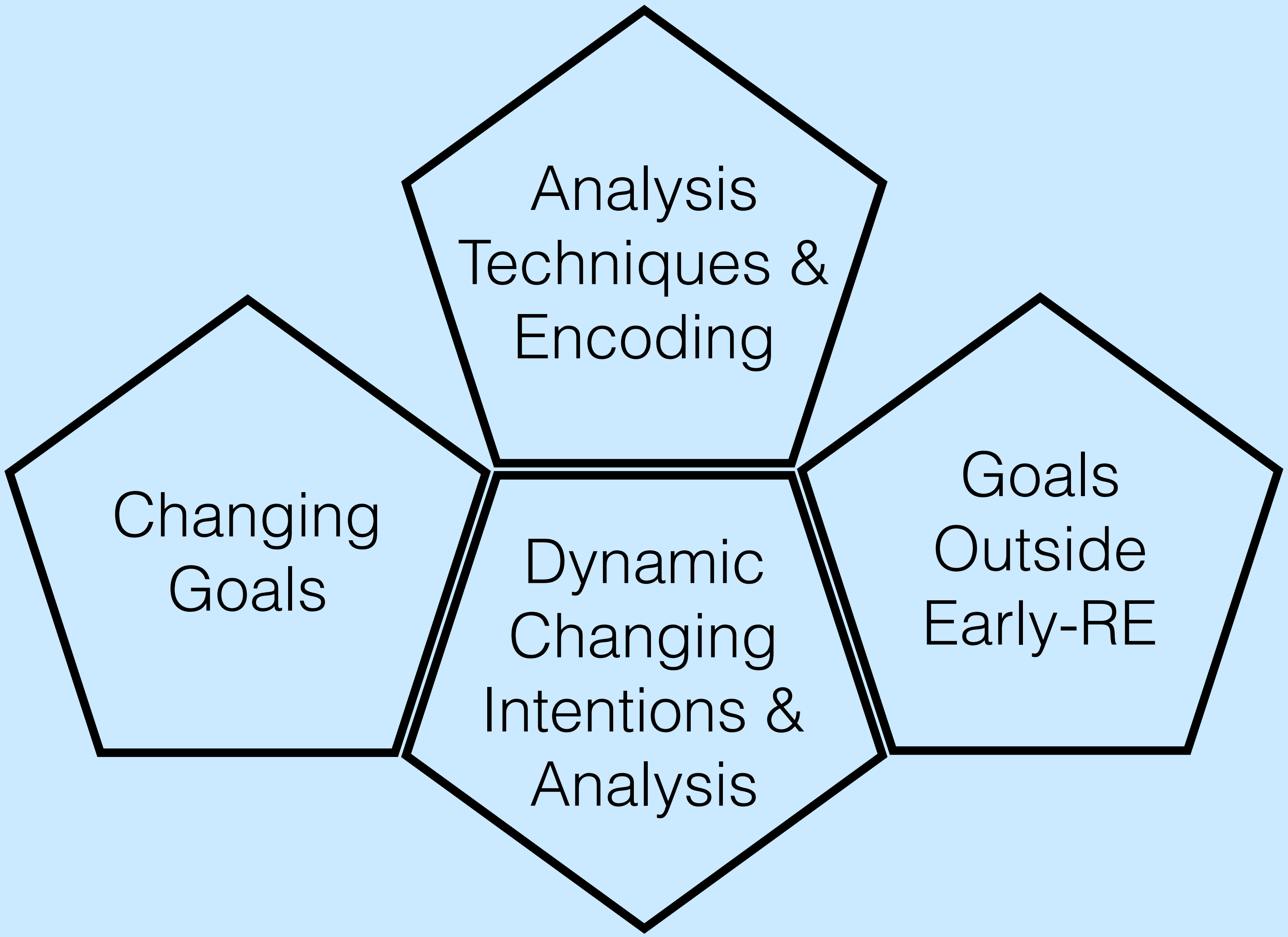
Related Work



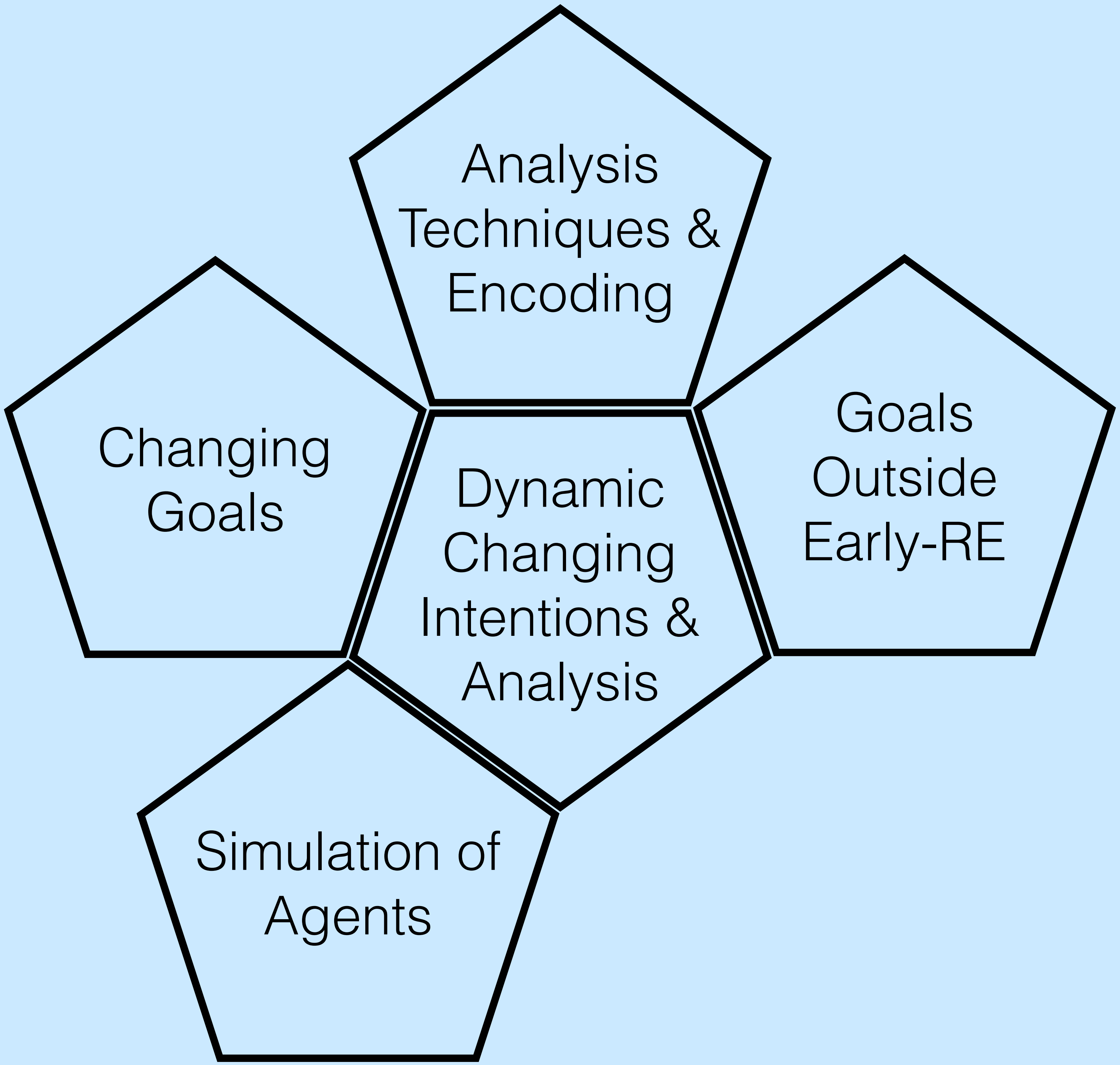
Related Work



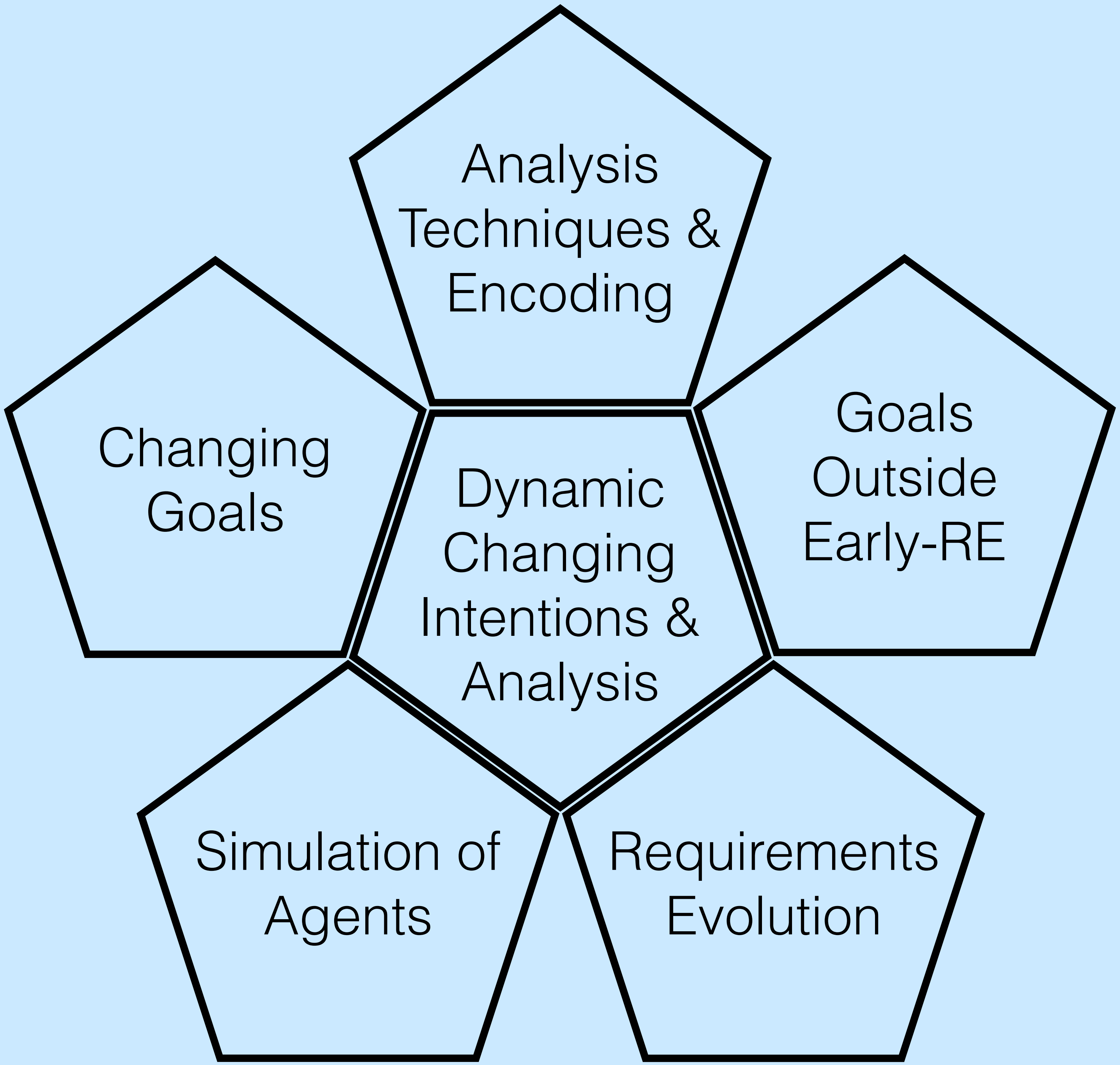
Related Work



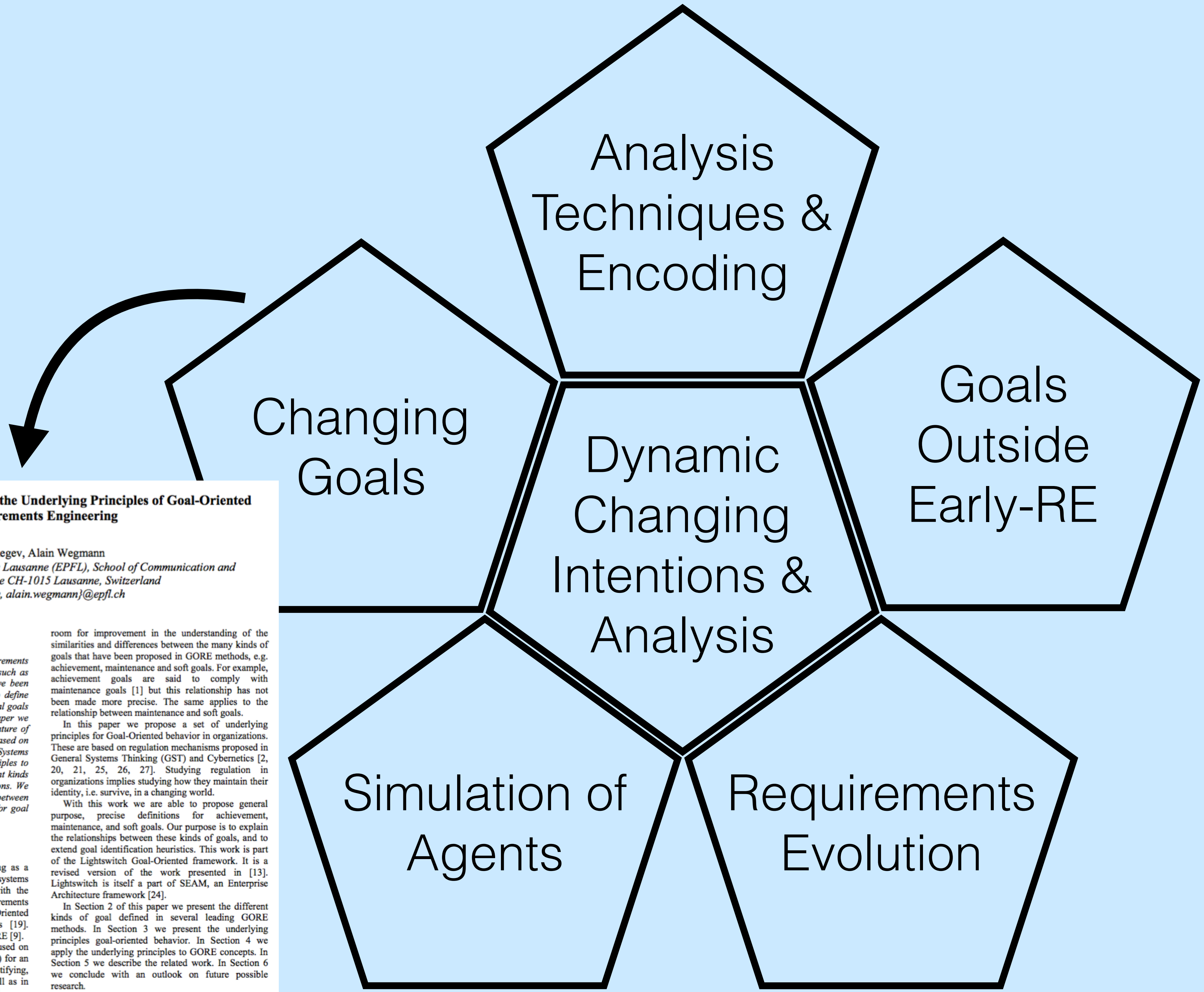
Related Work



Related Work



Related Work



Where do Goals Come from: the Underlying Principles of Goal-Oriented Requirements Engineering

Gil Regev, Alain Wegmann
Ecole Polytechnique Fédérale de Lausanne (EPFL), School of Communication and Computer Science CH-1015 Lausanne, Switzerland
{gil.regev, alain.wegmann}@epfl.ch

Abstract

Goal is a widely used concept in requirements engineering methods. Several kinds of goals, such as achievement, maintenance and soft goals, have been defined in these methods. These methods also define heuristics for the identification of organizational goals that drive the requirements process. In this paper we propose a set of principles that explain the nature of goal-oriented behavior. These principles are based on regulation mechanisms as defined in General Systems Thinking and Cybernetics. We use these principles to analyze the existing definitions of these different kinds of goals and to propose more precise definitions. We establish the commonalities and differences between these kinds of goals, and propose extension for goal identification heuristics.

1 Introduction

The emergence of requirements engineering as a separate discipline from computer science and systems engineering in the early 1990s coincided with the development of methods for defining requirements based on goals, the so called Goal-Oriented Requirements Engineering (GORE) methods [19]. Goals are now considered as a core concept in RE [9].

Requirements engineering research has focused on goals as a way of providing the rationale (why) for an envisioned system [18]. This helps in identifying, organizing, and managing requirements as well as in driving the requirements elaboration process [1].

Several GORE methods have been defined that give more attention to one or more of these aspects e.g. CREWS [15], GBRAM [1], GRL [8], i* [28], KAOS [7], TROPOS [11], etc.

GORE research has focused on the development of methods. Little research has been done on the underlying principles of GORE [9]. As a result there is room for improvement in the understanding of the similarities and differences between the many kinds of goals that have been proposed in GORE methods, e.g. achievement, maintenance and soft goals. For example, achievement goals are said to comply with maintenance goals [1] but this relationship has not been made more precise. The same applies to the relationship between maintenance and soft goals.

In this paper we propose a set of underlying principles for Goal-Oriented behavior in organizations. These are based on regulation mechanisms proposed in General Systems Thinking (GST) and Cybernetics [2, 20, 21, 25, 26, 27]. Studying regulation in organizations implies studying how they maintain their identity, i.e. survive, in a changing world.

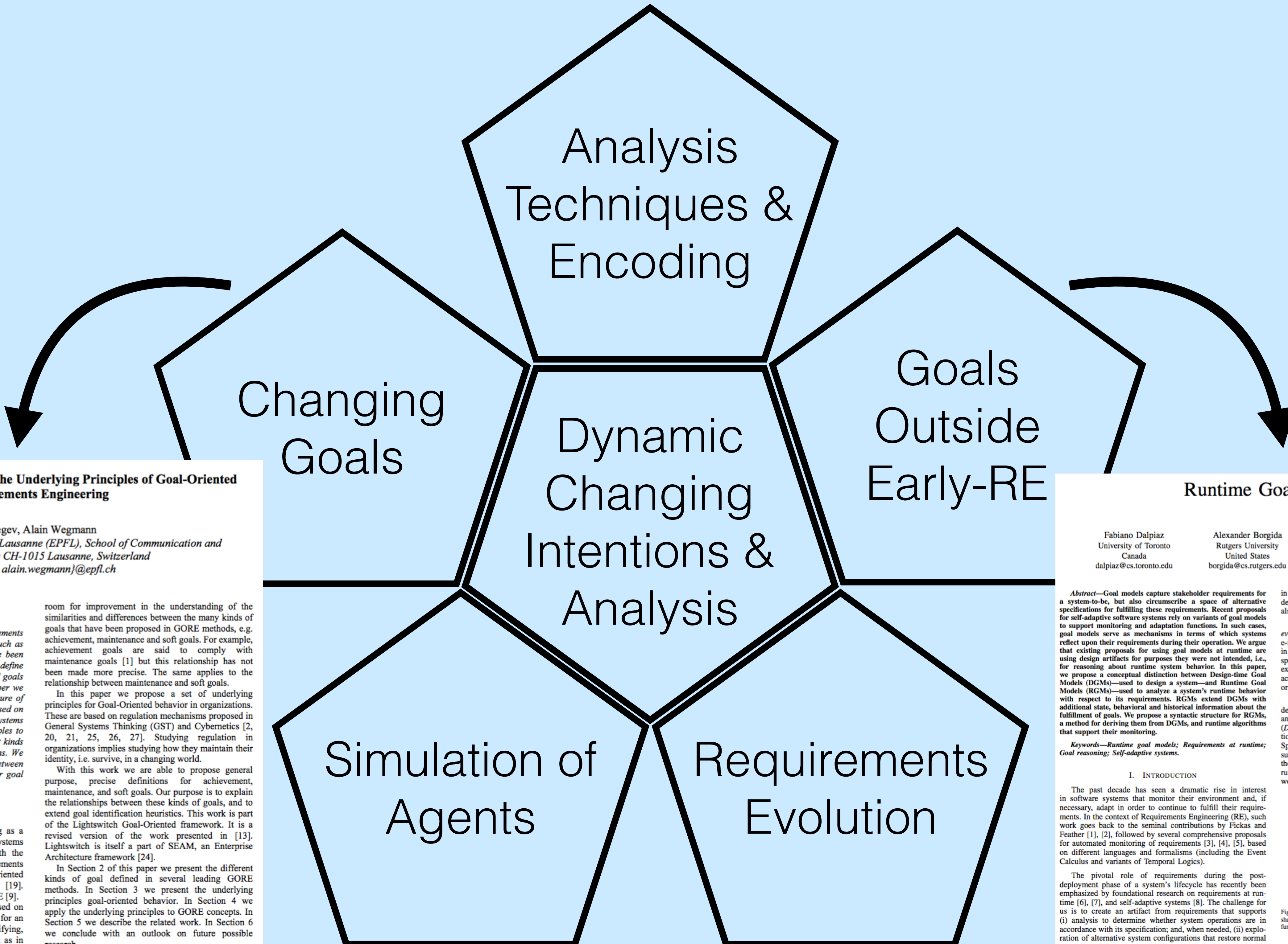
With this work we are able to propose general purpose, precise definitions for achievement, maintenance, and soft goals. Our purpose is to explain the relationships between these kinds of goals, and to extend goal identification heuristics. This work is part of the Lightswitch Goal-Oriented framework. It is a revised version of the work presented in [13]. Lightswitch is itself a part of SEAM, an Enterprise Architecture framework [24].

In Section 2 of this paper we present the different kinds of goal defined in several leading GORE methods. In Section 3 we present the underlying principles goal-oriented behavior. In Section 4 we apply the underlying principles to GORE concepts. In Section 5 we describe the related work. In Section 6 we conclude with an outlook on future possible research.

2 The Use of Goals in GORE methods

GORE methods take their root in AI research into problem solving [18]. The reasons for focusing on goals, found in the GORE literature [1, 8, 18], are the higher level view of requirements afforded by goals as

Related Work



Where do Goals Come from: the Underlying Principles of Goal-Oriented Requirements Engineering

Gil Regev, Alain Wegmann
Ecole Polytechnique Fédérale de Lausanne (EPFL), School of Communication and Computer Science CH-1015 Lausanne, Switzerland
{gil.regev, alain.wegmann}@epfl.ch

Abstract

Goal is a widely used concept in requirements engineering methods. Several kinds of goals, such as achievement, maintenance and soft goals, have been defined in these methods. These methods also define heuristics for the identification of organizational goals that drive the requirements process. In this paper we propose a set of principles that explain the nature of goal-oriented behavior. These principles are based on regulation mechanisms as defined in General Systems Thinking and Cybernetics. We use these principles to analyze the existing definitions of these different kinds of goals and to propose more precise definitions. We establish the commonalities and differences between these kinds of goals, and propose extension for goal identification heuristics.

1 Introduction

The emergence of requirements engineering as a separate discipline from computer science and systems engineering in the early 1990s coincided with the development of methods for defining requirements based on goals, the so called Goal-Oriented Requirements Engineering (GORE) methods [19]. Goals are now considered as a core concept in RE [9].

Requirements engineering research has focused on goals as a way of providing the rationale (why) for an envisioned system [18]. This helps in identifying, organizing, and managing requirements as well as in driving the requirements elaboration process [1].

Several GORE methods have been defined that give more attention to one or more of these aspects e.g. CREWS [15], GBRAM [1], GRL [8], i* [28], KAOS [7], TROPOS [11], etc.

GORE research has focused on the development of methods. Little research has been done on the underlying principles of GORE [9]. As a result there is room for improvement in the understanding of the similarities and differences between the many kinds of goals that have been proposed in GORE methods, e.g. achievement, maintenance and soft goals. For example, achievement goals are said to comply with maintenance goals [1] but this relationship has not been made more precise. The same applies to the relationship between maintenance and soft goals.

In this paper we propose a set of underlying principles for Goal-Oriented behavior in organizations. These are based on regulation mechanisms proposed in General Systems Thinking (GST) and Cybernetics [2, 20, 21, 25, 26, 27]. Studying regulation in organizations implies studying how they maintain their identity, i.e. survive, in a changing world.

With this work we are able to propose general purpose, precise definitions for achievement, maintenance, and soft goals. Our purpose is to explain the relationships between these kinds of goals, and to extend goal identification heuristics. This work is part of the Lightswitch Goal-Oriented framework. It is a revised version of the work presented in [13]. Lightswitch is itself a part of SEAM, an Enterprise Architecture framework [24].

In Section 2 of this paper we present the different kinds of goal defined in several leading GORE methods. In Section 3 we present the underlying principles goal-oriented behavior. In Section 4 we apply the underlying principles to GORE concepts. In Section 5 we describe the related work. In Section 6 we conclude with an outlook on future possible research.

2 The Use of Goals in GORE methods

GORE methods take their root in AI research into problem solving [18]. The reasons for focusing on goals, found in the GORE literature [1, 8, 18], are the higher level view of requirements afforded by goals as

Runtime Goal Models

Fabiano Dalpiaz
University of Toronto
Canada
dalpiaz@cs.toronto.edu

Alexander Borgida
Rutgers University
United States
borgida@cs.rutgers.edu

Jennifer Horkoff, John Mylopoulos
University of Trento
Italy
{horkoff, jm}@disi.unitn.it

***Abstract*—Goal models capture stakeholder requirements for a system-to-be, but also circumscribe a space of alternative specifications for fulfilling these requirements. Recent proposals for self-adaptive software systems rely on variants of goal models to support monitoring and adaptation functions. In such cases, goal models serve as mechanisms in terms of which systems reflect upon their requirements during their operation. We argue that existing proposals for using goal models at runtime are using design artifacts for purposes they were not intended, i.e., for reasoning about runtime system behavior. In this paper, we propose a conceptual distinction between Design-time Goal Models (DGMs)—used to design a system—and Runtime Goal Models (RGMs)—used to analyze a system's runtime behavior with respect to its requirements. RGMs extend DGMs with additional state, behavioral and historical information about the fulfillment of goals. We propose a syntactic structure for RGMs, a method for deriving them from DGMs, and runtime algorithms that support their monitoring.**

***Keywords*—Runtime goal models; Requirements at runtime; Goal reasoning; Self-adaptive systems.**

I. INTRODUCTION

The past decade has seen a dramatic rise in interest in software systems that monitor their environment and, if necessary, adapt in order to continue to fulfill their requirements. In the context of Requirements Engineering (RE), such work goes back to the seminal contributions by Fickas and Feather [1], [2], followed by several comprehensive proposals for automated monitoring of requirements [3], [4], [5], based on different languages and formalisms (including the Event Calculus and variants of Temporal Logics).

The pivotal role of requirements during the post-deployment phase of a system's lifecycle has recently been emphasized by foundational research on requirements at run-time [6], [7], and self-adaptive systems [8]. The challenge for us is to create an artifact from requirements that supports (i) analysis to determine whether system operations are in accordance with its specification; and, when needed, (ii) exploration of alternative system configurations that restore normal operations and continuing fulfillment of requirements. In this paper we address the first challenge, showing how to create a runtime artifact from requirements allowing for monitoring and diagnosis.

Some existing approaches (e.g., [5], [9], [10]) rely on early requirements models, often variants of *i**goal models [11], which are insufficiently detailed to effectively express (un)desired system behavior. Early requirements models talk about stakeholder goals and needed functionality for the system-to-be (e.g., sending e-mail), and can play a key role in communicating between various stakeholders by abstracting details that are irrelevant at this level. Requirement models are also useful for choosing among design alternatives.

However, at runtime, system behavior is characterized by *events occurring in the world*, related to goal *instances* (e.g., an e-mail has been sent from X to Y for purpose Z at some point in time). Some approaches (e.g., [4], [12], [13]) adopt low-level specification languages, resembling programming calculi, to express behavior. These are easy to monitor, for they capture actual program behavior, but are difficult to trace back to original stakeholder requirements.

We provide a framework for bridging the gap between design-time goal models and runtime behavior. Starting with an early requirements model representing stakeholder goals (*Design-time Goal Model* or *DGM*), we refine it with additional behavioral detail about how goals are to be achieved. Specifically, we add constraints on valid orderings for pursuing subgoals, thereby creating a *Runtime Goal Model* (*RGM*). Although this information helps us to express additional desired runtime behavior, this model is still at the class level, while we really need to reason over multiple goal instances.

Fig. 1. A partial DGM for an ATM system, adapted from [5]. Goals (oval shapes) are refined via AND/OR refinement links to tasks (hexagons), i.e., functionalities for the system-to-be

We elucidate the need to distinguish between classes and instances using the partial DGM for an Automated Teller Machine (ATM) system (Figure 1) adapted from [5]. The model is informative at the class level, as it conveys information about required functionality for the system-to-be. However, when used at runtime, it does not explain when and how many instances of the goals and tasks in the model need to be created, nor how many have been created. For example, putting in cash (task T_2) may be unnecessary for starting the ATM (achieving goal G_2), if enough money is already available. In such situation, no instance of T_2 is needed to achieve

Summary

Motivating Example

Goal: Evaluate waste management infrastructure

Intentions: Wants to be green and satisfy customer

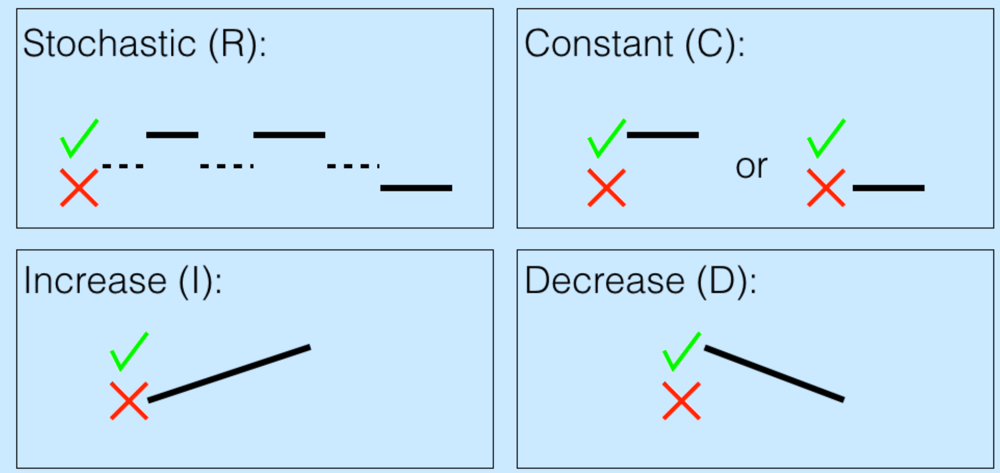
Options: Build Green Centre
Build Landfill / Dump (large, small)

Approach: Choose correct alternative(s)
using goal modeling.

2

Modeling Dynamic Intentions

Elementary Functions



14

Strategies

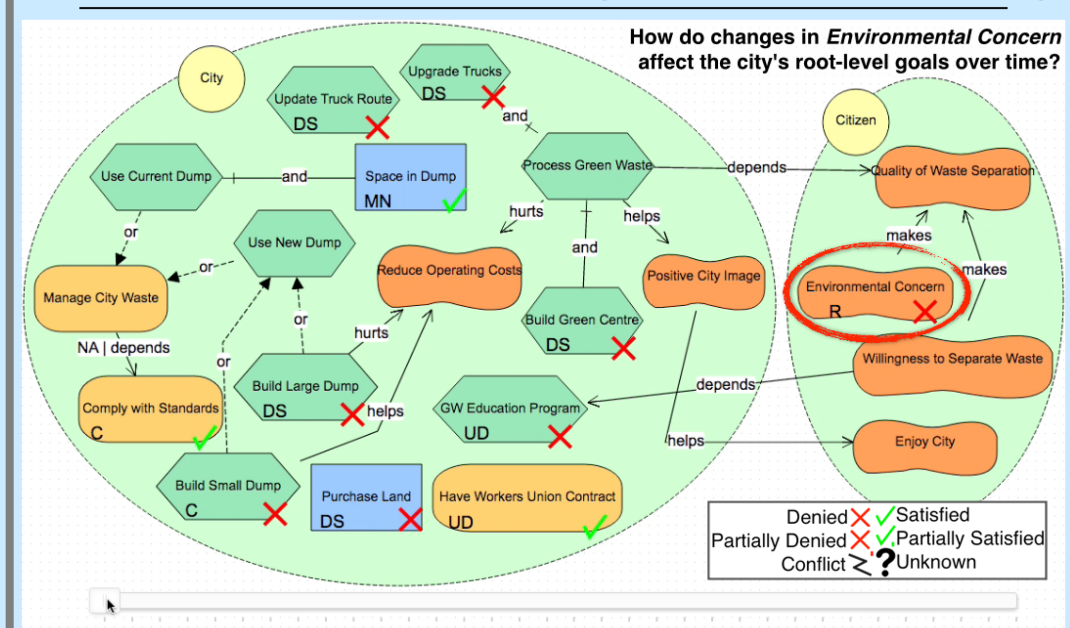
(Strategy 1) create a **random path** given initial states in the model

(Strategy 2) create a path given **desired properties** of the **intermediate state** (with optional properties over the initial or final state)

(Strategy 3) create a path which is **different than the previously seen path** over the same constraints

21

Leaf Simulation (Initial States)



R:Stochastic, C:Constant, UD:User Defined 24 DS:Denied-Satisfied, MN:Monotonic Negative

Future Work

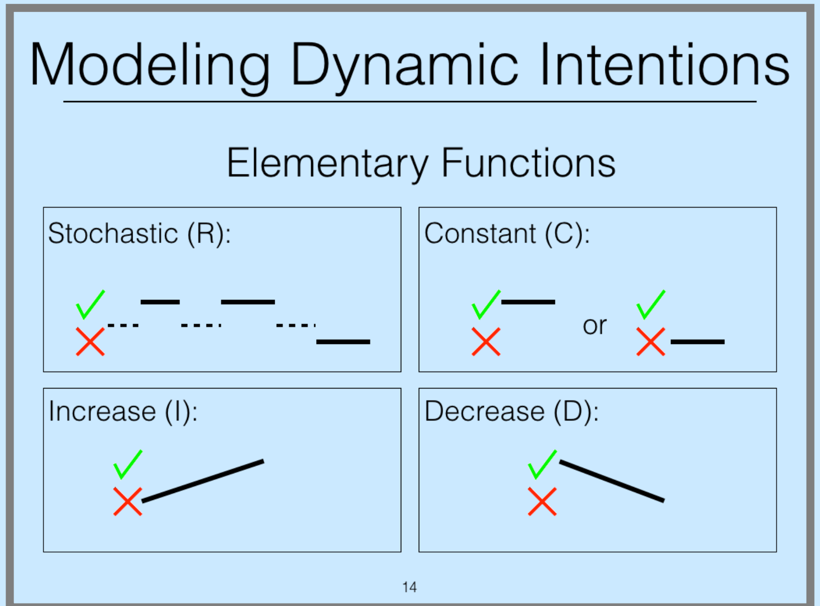
- Evaluate effectiveness
- External industrial case study
- Add “wall clock time” to analysis
- Optimize CSP encoding
- Formally specify our extension

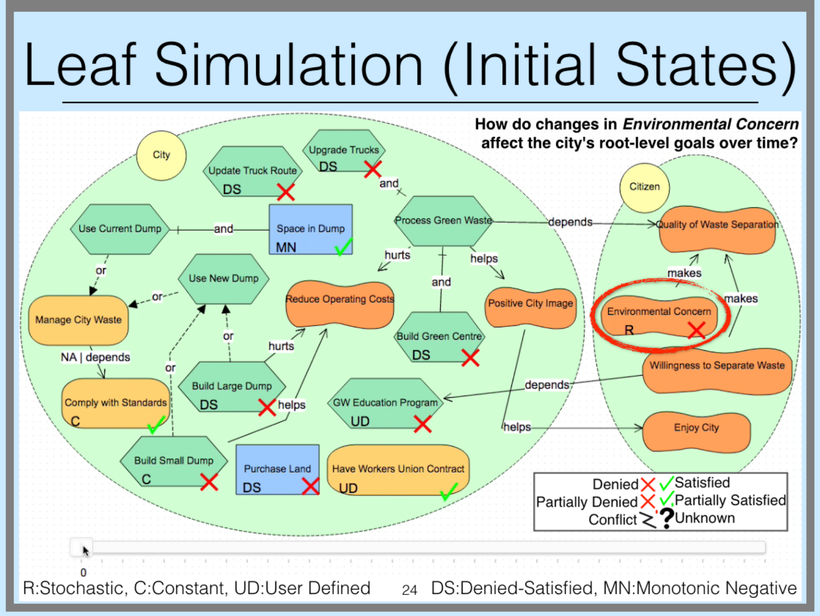
Questions?

Looking into the Crystal Ball: Requirements Evolution over Time

Contributions:

- Understand the impacts of dynamically changing intentions on decision making
- Enrich goal models intentions with dynamically changing evaluation





Alicia M. Grubb
amgrubb@cs.toronto.edu

UNIVERSITY OF TORONTO