Automated Reasoning About Classical Mechanics

by

Leon Chih Wen Wong

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1994

Author . . . .
Department of Electrical Engineering and Computer Science
May 5, 1994

Certified by . . . .
Howard E. Shrobe
Principal Research Scientist
Thesis Supervisor

Accepted by . . . .
Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students

Automated Reasoning About Classical Mechanics

by

Leon Chih Wen Wong

Submitted to the Department of Electrical Engineering and Computer Science
on May 5, 1994, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

Abstract

In recent years, researchers in artificial intelligence have become interested in replicating human physical reasoning talents in computers. One of the most important skills in this area is the ability to predict how physical systems behave. This thesis discusses an implemented program that can generate algebraic descriptions of how systems of rigid bodies evolve over time. Discussion about the design of this program identifies a powerful physical reasoning paradigm and knowledge representation approach based on mathematical model construction and algebraic reasoning. This paradigm offers a number of advantages over methods that have become popular in the field, and seems a promising approach for reasoning about a wide variety of classical mechanics problems.

Thesis Supervisor: Howard E. Shrobe
Title: Principal Research Scientist

Automated Reasoning About Classical Mechanics

by

Leon Chih Wen Wong

Submitted to the Department of Electrical Engineering and Computer Science
on May 5, 1994, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

Abstract

In recent years, researchers in artificial intelligence have become interested in replicating human physical reasoning talents in computers. One of the most important skills in this area is the ability to predict how physical systems behave. This thesis discusses an implemented program that can generate algebraic descriptions of how systems of rigid bodies evolve over time. Discussion about the design of this program identifies a powerful physical reasoning paradigm and knowledge representation approach based on mathematical model construction and algebraic reasoning. This paradigm offers a number of advantages over methods that have become popular in the field, and seems a promising approach for reasoning about a wide variety of classical mechanics problems.

Thesis Supervisor: Howard E. Shrobe
Title: Principal Research Scientist

Acknowledgments

I would like to thank Thomas Stahovich and Randall Davis for their valuable help with many aspects of my research. Special thanks go to Howard Shrobe, my supervisor, for his insight and guidance over the course of this project.

This research was conducted at the Massachusetts Institute of Technology Artificial Intelligence Laboratory. It was supported by the Advanced Research Projects Agency of the Department of Defence under Office of Naval Research contract number N00014-91-J-4038.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Much of human behavior involves interactions with the physical world. From space flight to microelectronics, some of humanity's finest intellectual achievements have been directed toward understanding our physical environment and manipulating it for an amazingly diverse number of different purposes.

It is hardly surprising, therefore, that researchers in artificial intelligence should seek to replicate human physical reasoning skills in computers. With such abilities, computers might provide intelligent assistance to engineers, interact more effectively with their physical environment, and better understand the physical metaphors that underlie much of human cognition.

Physical reasoning can be separated into two broad categories: design and analysis. Analysis represents the ability to understand how physical systems behave, while design represents the ability to create physical systems that act in some desired fashion. Both exist as areas of active interest in the artificial intelligence community; however, much of the work to date, including this thesis, has focused on the analysis portion of the task. The reason is quite simply that in human experience, understanding physical systems has been essential to effectively manipulating them to meet our goals.

In the analysis of physical systems, computers have traditionally played the limited role of performing numerical simulation from mathematical models. While this has been immensely useful, computers have been much less successful at higher level tasks such as creating the mathematical models to simulate, and interpreting the results. Furthermore, there are many cases where numerical reasoning is inappropriate or impossible: for example, situations where information is incomplete. Such cases are common in the early stages of design, and when parameters of real world systems are difficult to measure. A related drawback to numerical reasoning is that the results are too specific to generalize in any way.

In the quest to extend computers' physical reasoning skills, researchers have used the abilities of human engineers as a source of inspiration. Engineers are physical reasoning specialists: they combine common sense intuitions, formal training, and expertise from practical experience to attack the complexity of physical systems. Examples of human skills that artificial intelligence researchers have sought to emulate include:

- Reasoning at multiple levels of abstraction
- Accommodating incomplete information
- Constructing models of physical systems that simplify reasoning by describing properties relevant to the task at hand, but at the same time hiding irrelevant complexity.
- Solving problems efficiently by adapting solutions from past experience.

In addition to examining the reasoning paradigms used by engineers, researchers have also sought to duplicate the physical domain knowledge that people bring to bear on physical reasoning tasks. In particular, many have studied physical intuitions: those skills that allow even people with no formal training to interact effectively with their physical environment. In addition to their power in supporting everyday activities, this kind of common sense reasoning often guides engineers, the physical reasoning elite, in their application of more formal methods of analysis.

1.2 Project Goals

The goal of this thesis was to investigate methods for solving analysis problems in classical mechanics. There were several reasons behind this choice of problem domain. One of the main motivations was that classical mechanics explains a large portion of the physical phenomena that people encounter in their regular activities. For this reason, methods for analyzing problems in this domain might offer a wide range of potential applications.

For the same reason, people have highly developed intuitions about mechanics that help them understand other physical sciences. This suggests that studying classical mechanics might offer insights into reasoning effectively about other domains. Yet another reason for examining this domain is that as a result of centuries of study, people understand classical mechanics exceptionally well. This allows research to focus on issues solely related to transferring an understanding of the domain to computers. Lastly, despite the fact that mechanics scenarios are relatively simple to describe, they are capable of producing an interestingly complex array of behavior that challenges the state of the art.

Given the focus of this thesis on classical mechanics, my research has specifically addressed three elements:

- Formalizing the knowledge required to understand classical mechanics.
- Representing this knowledge in an effective manner
- Developing inference techniques that can effectively apply represented domain knowledge to solving analysis problems

One of the primary difficulties in investigating these aspects remains the fact that, despite the wealth of formal methods for analyzing the domain, much of human performance depends on poorly understood intuitions. Some of the challenges in automating classical mechanics reasoning therefore include substituting appropriate computer knowledge and inference methods for human physical intuitions, and interfacing these with the field's traditional formal methods.

Complicating this effort is the fact that computers display a very different set of strengths and weaknesses from people. This opens the door to certain powerful approaches, such as computationally intensive algebraic techniques, but also complicates efforts to duplicate many human problem solving faculties.

1.3 Thesis Organization

This document describes research efforts toward the goals presented in the previous section. The next chapter describes AMES (Algebraic MEchanics Simulator), an implemented program that can reason about a limited range of classical mechanics problems. The chapter that follows distills the key features from AMES' design and illustrates how these principles might be generalized to expand the system's reasoning capabilities. This discussion also highlights many of the strengths and weaknesses of AMES' approach to physical analysis.

Chapter 4 builds on this evaluation of AMES' abilities by outlining some areas for future research. Following this, the last two chapters of the thesis look at how my research relates to other work in the field, and provide some concluding remarks.

Chapter 2

AMES System Description

2.1 Introduction

This chapter presents a description of AMES (Algebraic MEchanics Simulator). AMES is an implemented software system that can predict the behavior of a small range of scenarios from classical mechanics. It was designed to explore issues in effectively capturing and applying knowledge about this domain.

The next section describes the specific tasks and issues that AMES was designed to address. Section 2.3 provides an overview of the major features of AMES' design. Section 2.4 demonstrates how these elements interact in analyzing an introductory example. Following this is a detailed description of the system's architecture. Finally, the chapter ends with some examples that illustrate the range of AMES' capabilities.

2.2 Task Description

The previous chapter explained that the goal of this research project was to study knowledge, representations, and inference methods for reasoning about classical mechanics. The AMES program addresses these objectives by focusing on a single task: generating descriptions of how classical mechanics systems evolve from specifications of their initial conditions. This simulation problem was ideal for two reasons. First, understanding how systems evolve over time is a fundamental component of most reasoning tasks in classical mechanics. Second, although simulation requires thorough knowledge of the domain, the problem model itself is relatively simple and constrained.

Note that two features differentiate AMES from numerical simulators, despite the fact that they share behavior prediction as their objective. First, AMES' input consists of high level information appropriate for discussing classical mechanics systems: the objects in the scenario, and the values of their various static and dynamic attributes like shape, velocity, and field strength. AMES must therefore understand the relationship between system characteristics and the behaviors they generate. Numerical simulators, on the other hand require mathematical models and initial variable values as input. The user must do all the reasoning about behavior in the domain to create these models; therefore, the simulators themselves have no physical knowledge.

The second major difference from numerical simulators is that AMES was designed to accommodate information at a level of detail comparable to that found in introductory mechanics textbook problems. In particular, AMES handles scenarios described using algebraic, as opposed to numerical quantities. This is interesting from a research perspective for the same reason that educators teach the material in this fashion. Algebraic quantification accommodates a degree of incomplete information, allows one to generalize over ranges of parameter values, and produces not only solutions, but also the factors on which they depend. The ability to use algebraic, as opposed to numerical reasoning, therefore, appears important to emulating some of the human skills that have inspired researchers in physical reasoning.

Due to the limited resources for this project, however, AMES reasons about only a narrow subset of the classical mechanics domain: frictionless two-dimensional rigid bodies with no rotational degrees of freedom. Furthermore, although the system detects collisions, it does not predict their consequences. Lastly, AMES cannot accommodate ambiguity in the possible behaviors a physical system might exhibit. This kind of situation can occur whenever there is insufficiently detailed information about the values of the parameters of physical systems.

While the scope of these abilities is fairly narrow, this problem domain was still sufficiently complex to raise issues of very general interest. Subsequent chapters will outline how the principles learned from this simple system's design can be extended to overcome many of the prototype's defects and limitations.

The following two sections outline the key elements of AMES' architecture, and present a brief illustration of how they work together to solve a simple mechanics problem. Section 2.5 elaborates on this with a much more detailed description of the system's reasoning, and more complex examples.

2.3 Design Overview

This section describes the high level organization of the Algebraic MEchanics Simulator (AMES). As mentioned in the previous section, AMES was designed to predict the behavior of mechanics systems from physical descriptions of their initial configurations.

AMES shares the same general reasoning paradigm as numerical simulators. It predicts the behavior of physical systems incrementally: information about each successive time interval in a scenario's evolution comes from the analysis of its predecessor. A unique feature about AMES, however, is that the granularity of simulation time intervals is much more coarse that those in numerical simulation. While numerical simulators reason about the changes that occur over very small fixed length time intervals, AMES reasons about changes that occur over longer variable durations of time called *qualitative states*.

In AMES, a *qualitative state* is a period of time over which one can describe the evolution of a physical system using a single mathematical model. This notion is therefore sensitive to the power of the mathematical infrastructure supporting the simulator, as well as the fundamental complexity of each physical system's behavior.

The reason such states are termed "qualitative" in nature is that changes in what people recognize as the high level behaviors of a physical system typically translate into changes in the mathematics required to model them. For example, one might informally judge static and sliding friction to be qualitatively distinct behaviors. AMES' knowledge-base would represent a similar distinction, though the criterion would be more objective: a transition between the two conditions implies a change in the mathematical description of frictional force. For the static case, friction balances all other forces in the direction parallel to the contact generating the force, in the contact frame of reference. Sliding friction, however is proportional to contact normal force, and is directed opposite the direction of relative motion.

The definition of qualitative states in terms of mathematical models is more than an attempt at formalizing informal notions of qualitative distinctiveness, however. The main reason for this manner of partitioning simulation histories is that AMES constructs mathematical models to reason about the behavior of physical systems. It is not always the case, however, that a single model can describe the entire evolution of a system. Therefore, AMES must partition its envisioning into intervals over which individual models hold.

Given this kind of reasoning scheme, the central element in AMES' simulation procedure is naturally the individual qualitative state analysis: the process by which AMES models a system's behavior during a state, reasons about how that state ends, and generates information to support similar analysis of its successor. Each successive application of this procedure pushes the simulation another qualitative state further into the future. Note that although AMES operates at a higher level of abstraction, this organization of reasoning is very similar to that found in Qualitative Simulation [21]: a technique for qualitatively solving systems of differential equations.

As was just mentioned, the state analysis process begins with constructing a mathematical model that describes how attributes of a physical system will evolve over the course of that state. Model construction proceeds in several stages. Figure 2.3 illustrates the organization of the process. The schematic reflects both the structure of domain knowledge within AMES, and the flow of information during model assembly.

Each module in the state analysis examines a different aspect of a physical system, and each can generate information of three different types. Modules may conclude qualitative information about the current state: for example, that contact between two objects exist. Modules can also generate quantitative information: sets of con-

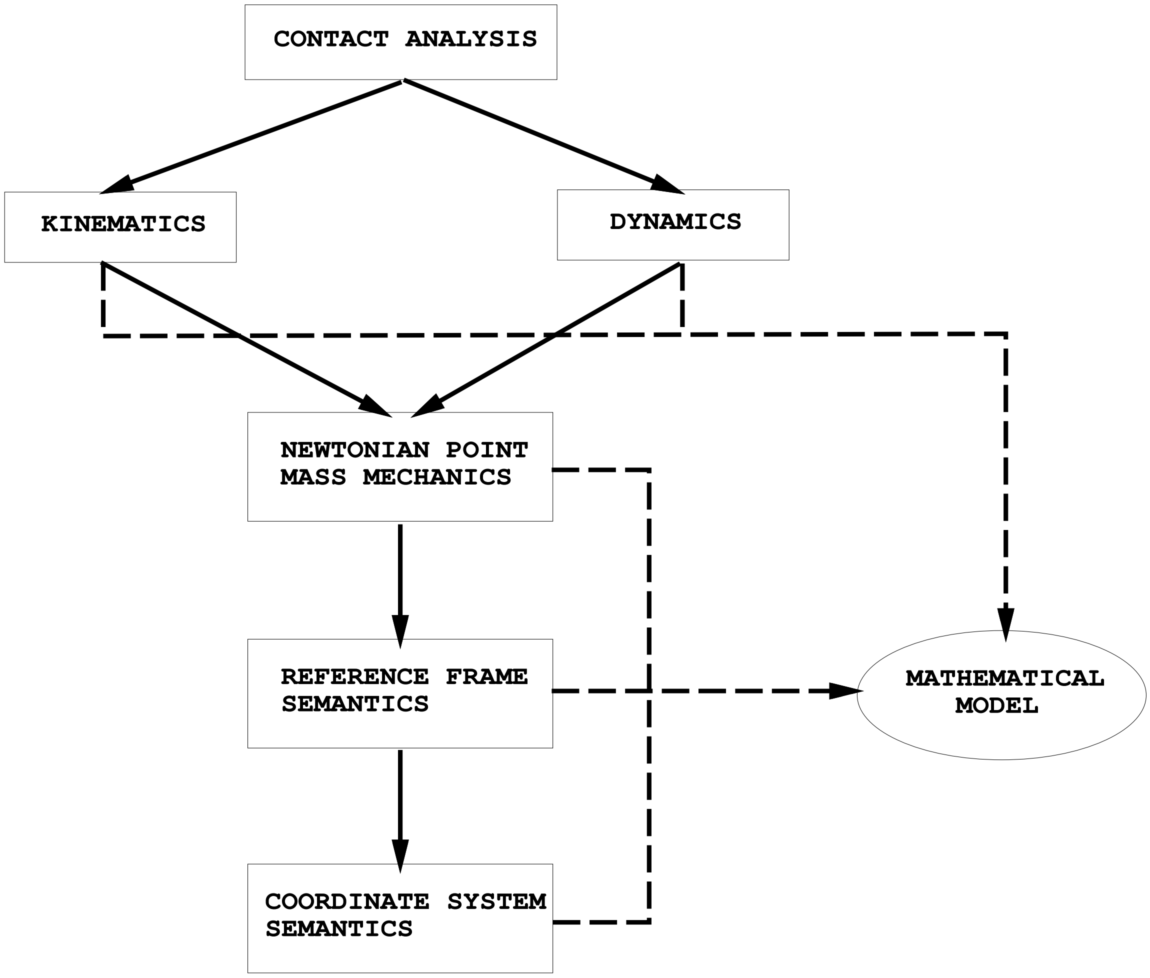

Figure 2-1: Components of State Analysis

Module	Description
Contact Analysis	Determines the contacts in the scenario
Kinematics	Determines restrictions on bodies motions
Dynamics	Describes the forces in the scenario
Point Mass Mechanics	Applies Newton's laws, and relates motion attributes to each other
Reference Frame Semantics	Relates quantities measured in different reference frames
Coordinate System Semantics	Relates quantities measured in different coordinate systems

Table 2.1: Module Descriptions

straints to incorporate into the mathematical model of the state. Finally, modules may generate state termination conditions: specifications of the events that would force changes to the model. Table 2.3 summarizes the role of each analysis module in AMES.

Once AMES generates a mathematical model of a qualitative state, the next step is to determine when and how the state ends. As previously mentioned, during the model construction phase, analysis modules generate state termination conditions that indicate when each part of the model contributed by those modules becomes invalid.

AMES therefore detects the end of the state by simply solving for when, if ever, the first state termination condition occurs. Then, AMES uses the current model to solve for the initial configuration of the subsequent state. The analysis of the next qualitative state uses these initial conditions along with information about the transition that created the state change as input.

The next section outlines the how AMES follows the above pattern of reasoning to analyze a simple physical scenario. The section after that explores the details of AMES' operation.

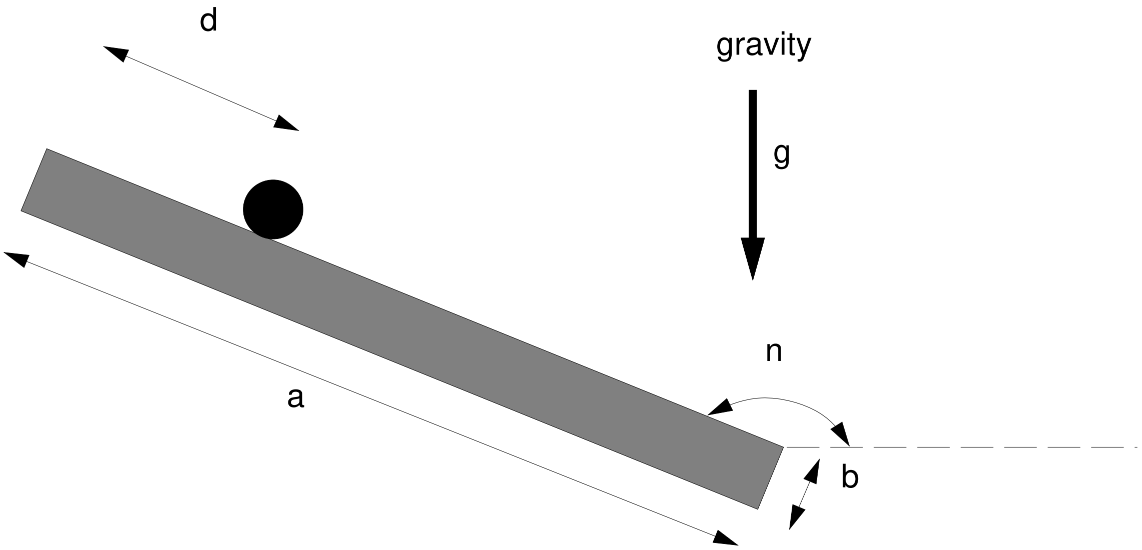

Figure 2-2: Particle on an Inclined Plane

2.4 Introductory Example

This section gives an overview of how AMES predicts the behavior of a particle initially at rest on an inclined plane. Figure 2.4 depicts the scenario. In addition, the following constraints apply: $0 < d < a$, $\frac{\pi}{2} < n < \pi$, $g > 0$, $mass_{particle} > 0$, and $velocity_{plane}(t) = 0$. Note that AMES requires more complete information about its input scenarios than a human would, since it does not know how to make reasonable assumptions about missing information, and lacks the knowledge to interpret implicit information contained in diagrams.

AMES' analysis of this problem begins by searching for all the contacts that exist in the initial state. It detects only the contact between the particle and the top of the inclined plane from information about their initial positions, and the fact that the contacting bodies have no initial velocity away from each other. In the remainder of its analysis of the initial state, AMES can assume that this contact will persist, since in the world of rigid body interactions, contact configuration completely describes qualitative states. Therefore, breaking the contact would create qualitative state change.

Note that since AMES can parameterize over different positions of the particle on the top of the plane, it can use the same model for reasoning about the scenario as long as the particle contacts that particular side of the incline. The range of validity of the model also determines the scope of the qualitative state, by definition.

From the contact configuration, AMES can deduce information about the state's kinematics and dynamics. Since the state lasts as long as the contact is present, it is possible to add equations to the model that establish that the particle's velocity will be tangential to the top of the incline. The usefulness of this information resides in the way it constrains the value of the normal forces between the bodies.

In analyzing the situation's dynamics, the existence of the contact establishes the existence of contact forces. From AMES' knowledge about these forces, it asserts that their direction is perpendicular to the contacting surfaces. The magnitude of the contact forces comes indirectly from the kinematic constraint: they take on whatever value is necessary to keep the particle moving tangentially to the surface of the incline. This is AMES' method for defining the behavior of compensating forces.

Note, however, that this model is not always accurate: contact normal forces only repel. Therefore, if there were a force that pulled the two bodies apart, the normal force could not resist. The particle would acquire velocity away from the incline, and therefore not move tangentially to its surface, producing a contradiction. This is where reasoning about model limitations enters. AMES handles the potential problem by having a termination condition that states that the model must change if it predicts attractive contact normal forces.

The other element to the system's dynamics, aside from the contact forces, are the gravitational forces. AMES has knowledge that gravity acts on each rigid body in the scenario with a value equal to the product of the field's strength and each body's mass.

At this time, AMES has all behaviors in the scenario reduced to information about forces, and constraints on bodies' motions. It can therefore proceed to apply Newton's laws by relating action-reaction force pairs, and performing free body analysis of each body. This stage of analysis also adds to the model information about the derivative relationships between positions, velocities and accelerations.

The last step in the model construction process is to add reference frame and coordinate system conversions. These are necessary because AMES' model typically has multiple variables representing different ways of measuring the same attributes. This apparent redundancy is useful because it is very convenient to discuss contacts, for example, in terms of relative motions between contacting bodies in parametric coordinates, while it may be more natural to discuss other behaviors, such as Newton's second law, in terms of inertial reference frames and cartesian coordinates.

When the modeling process is complete, AMES turns to reasoning about state change. The mathematical model it generated is only valid as long as the same contact configuration persists. AMES therefore has two types of conditions on the duration of the initial qualitative state: conditions that establish that no new contacts occur, and conditions that establish how long the existing contact persists. In this case, it is clear that no new contacts occur, since the only other possible contacts are between the particle and the other sides of the incline: these cannot happen since the sides occupy mutually exclusive portions of space. The old contact, on the other hand has 3 potential ways to end: the particle can move off the end at either the top or the bottom, or it might fly off in the middle.

The first two conditions are constraints on position, while the normal force magnitude constraint that was mentioned earlier implements the last condition. In the case of this scenario, AMES has enough information to solve for the fact that the particle in fact slides off the bottom of the incline. The state change therefore involves an end to the contact, and the next state has an empty contact configuration.

The analysis of the next state therefore concludes that the particle is in free fall. Furthermore, the direction of the particle's motion is such that new contacts are impossible. Therefore, the free fall state never terminates and the simulation is complete.

2.5 Detailed System Description

2.5.1 Overview

This section describes AMES' simulation procedure in detail. Note, however, that the implemented system queries the user for all mathematical reasoning tasks. The reason for this was that automated algebraic reasoning is fairly well understood, and therefore efforts in this area were unlikely to further this project's research goals. To ensure that the system's performance is still convincing, however, AMES has its own representation of mathematical objects, and all quantitative reasoning occurs through a narrow interface: either solving for the value of a variable or determining the truth of an expression.

The description of AMES in this section consists of several subsections. Each subsection illustrates the operation of a different component of the analysis procedure. The first several describe the different model construction modules. They each discuss the following aspects of the modules' operation:

- The aspect of physical scenarios that the module examines.
- The inference techniques used to perform the analysis.
- The contributions to the mathematical model of the state.
- The conditions on the validity of the analysis.

After the descriptions of the individual model construction modules, this section discusses how AMES uses its mathematical models of qualitative states to provide information for the analysis of their successors. The section closes with a summary of AMES knowledge about mechanics.

2.5.2 Contact Analysis

Scope

As the name implies, the contact analysis module produces a description of the various contacts between rigid bodies in the current state. This information is critical since it allows the system to deduce the presence of normal forces and various motion constraints. In AMES' limited domain, changes in contact configuration completely determine changes of qualitative state, since the program assumes that all the other highest level behaviors, namely gravitation and externally constrained factors, remain constant over the course of simulations.

When a contact exists between two objects, the contact module computes the locus of relative positions that allow the bodies to remain in contact. The shape of the contact locus, in turn, allows other parts of the state analysis to find the direction of surface normal forces, and the range of motions that the impenetrability of rigid bodies allows.

A very natural way to represent contacting positions is to compute the shapes of objects in other objects' *configuration spaces* [23]. This method reduces the problem of finding contacts in AMES' domain to the geometric problem of determining whether a point lies on the edge of a two-dimensional shape. Another advantage of this scheme is that features such as normal force directions remain unchanged by the configuration space transformation.

Note, however, that the outlines of shapes typically found in mechanics problems are often discontinuous: for example the discontinuities at the corners of a rectangular block. This prevents straightforward mathematical characterization of the entire locus of contact positions, and therefore complicates the mathematical modeling task. AMES' solution, therefore, partitions sets of contacting positions into piecewise simple segments, where "simple" is defined by the ability of the geometric analysis module to produce purely mathematical descriptions of each resulting shape. The rest of this thesis will refer to these subsets of contacting positions as *simple contacts*.

The contact module therefore describes the set of simple contacts between rigid bodies in each qualitative state. Note that unlike many of the other analysis modules, the contact analysis makes no direct contribution to the mathematical model. Instead, it provides information to the kinematics and dynamics modules. They, in turn, interpret the consequences of the contacts in terms of restrictions on bodies' degrees of freedom, and normal forces between contacting objects.

Inference Methods

AMES employs a straightforward technique for finding simple contacts and computing their shape. Before the simulation begins, AMES generates descriptions of all possible simple contacts that can occur between the rigid bodies in the problem scenario. During the simulation, the contact analysis module describes contact configurations by maintaining sets of pointers to the simple contacts that are actually present during each state. AMES actually needs information on all possible contacts since accurate modeling depends on knowing not only the current contact configuration, but also what new contacts might appear.

To generate all the possible contacts, AMES considers every possible pairing of rigid bodies. For each pair, it selects one object to be the "observer". The other object becomes the "obstacle". Then, AMES computes the shape of the obstacle in the observer object's configuration space. This gives the shape of the locus of contacting relative positions for that pair of objects.

Once the obstacle configuration space shape has been computed, the outline of the shape is decomposed into simple segments: shapes for which the quantitative reasoning engine can find mathematical descriptions. The current mathematics module supports two types of one-dimensional shape primitives: circular arcs and line segments. Therefore, all configuration space obstacle shapes are decomposed into these. Note that the decomposition is always possible since all of the rigid object shapes are also composed entirely from these primitives. Figure 2.5.2 gives a graphical summary of the contact generation process.

A small subtlety of this process is that the simple contact shapes are stored as directed paths. The direction information allows AMES to record which side of the

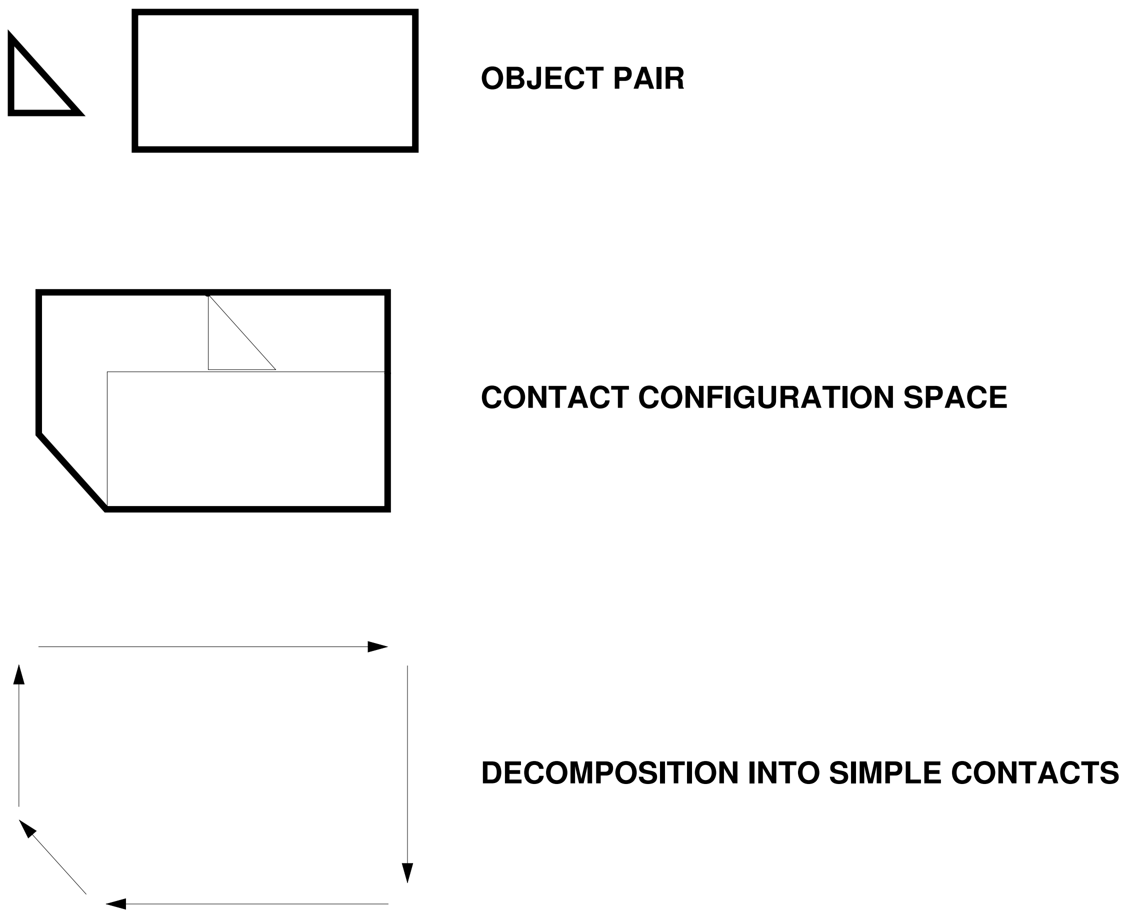

Figure 2-3: Summary of Contact Generation

path corresponds to the exterior of the obstacle object. This information is critical in determining the direction of normal forces between the contacting bodies.

An additional issue in the contact generation is choosing observer and obstacle roles for each pair of objects. This choice is actually unimportant in terms of producing a correct analysis, since the configuration space computation is commutative [23]. AMES does, however, use a heuristic to assign the role of "obstacle" to the object whose shape most resembles its configuration space shape: it makes the smaller object the "observer". This makes the analysis somewhat easier for human users to follow.

During the simulation, AMES checks for the existence of various contacts by considering the configuration space position of each observer object in the reference frame of their corresponding obstacles. If the position happens to be on a simple contact, then the system deduces that that contact appears in the state. Between states, if there is no evidence that a contact has been broken, then AMES simply carries the contact into the next state: there is no need to repeat the analysis. Similar reasoning also applies to predicting the contacts that are not present.

Dependencies

For a particular set of contacts to remain an accurate description of a state, two facts must hold. First, no existing contact can break, and second, no new contact can appear. The validity of the contact analysis therefore depends on formal descriptions of these two conditions. New contacts are easy to detect during the simulation, due to the extensive processing during the initialization phase: the configuration space shapes that are generated before the simulation begins indicate the relative positions of objects that result in contacts. During the simulation, therefore, satisfying the existence criterion of a simple contact that does not appear in the current state invalidates the state's contact description and leads to a new state.

Detecting when existing contacts disappear, however, is slightly more difficult. Every simple contact is a finite length one-dimensional path consisting of positions that the observer can occupy and still preserve contact. There are therefore two different ways to break an existing simple contact: the observer object can move beyond an endpoint of the contact path, or the observer can move off the path from some internal point.

The first condition is simple to express mathematically, especially since AMES uses distances along simple contact paths to describe displacement. Expressing the second condition turns out to be slightly more involved, however, because of the difficulty of expressing rigid bodies' tendency to slide against each other. To describe that behavior, the kinematics module, as will be described below, has a model where contacting bodies must always stay connected. This means that the state's mathematical model does not permit velocities to have components normal to the simple contact shape.

The actual state termination condition is therefore expressed in terms of contact normal force direction. If a contact force must be attractive in order to preserve contact, then contact breaks. The section on dynamics analysis will explore the rationale behind this design in greater detail.

2.5.3 Kinematics Analysis

Scope

The concept of impenetrability is a key element of informal descriptions of rigid bodies. This characterization is incomplete in many ways, however. In particular, it says little about what mechanism prevents bodies from occupying the same space. AMES provides this missing information in the kinematics and the dynamics that it associates with contacts.

The goal of the kinematics analysis is to refine higher level knowledge about qualitative states into constraints on objects' motions. Within the scope of problems that AMES addresses, this task translates into converting contact information and user specified motion restrictions into constraints on rigid body positions. AMES represents such degree of freedom restrictions with shapes that indicate the loci of positions that bodies can possibly occupy. These positions might be measured relative to any reference frame. This gives the description format the flexibility required to clearly describe the relative position constraints that arise from contacts.

Inference Methods

The kinematics module collects degree of freedom restrictions from two sources: user-supplied information about external influences on the scenario (like the "glue" that attaches things like floors and walls to the fixed frame), and the state's contact configuration. In the first case, no special inference is required, since the information comes directly from the user.

Deducing degree of freedom restrictions from contact information is not much more complex. For each simple contact in the current state, it must be the case that as long as that contact persists, the bodies must have relative positions inside the configuration space shape that describes the contact. Therefore, the degree of freedom restriction has the same shape as the contact locus.

Model Contribution

The kinematics module adds a set of equations to the mathematical model of the state for every degree of freedom restriction present. In AMES's world of rotation-free two-dimensional geometry, there are only two types of degree of freedom restrictions: zero-dimensional and one-dimensional.

In both cases, the motion constraints can be expressed as restrictions on objects' velocity. Zero-dimensional degree of freedom restrictions imply zero velocity, while one-dimensional degree of freedom restrictions imply that velocity must always be tangential to the path restriction: otherwise the object would move off the path. Note that the reason that AMES uses velocity constraints is that the equations tend to be simpler to express than the more fundamental position constraints. With the proper initial conditions, however, the two formulations are equivalent.

Dependencies

AMES assumes user-specified phenomena persist over the entire course of a simulation, therefore there are no state transition conditions to associate with these. For kinematic constraints from contacts, the conditions that guarantee contact are sufficient to ensure the correctness of the kinematic analysis.

2.5.4 Dynamics Analysis

Scope

The dynamics analysis determines the forces that act in the current state, and infers constraints on their values. AMES scenarios can contain three types of forces:

- Gravitational forces: the effects of gravitational fields on rigid bodies.
- Contact normal forces: the repulsive forces that prevent penetration and deformation of rigid bodies.
- External forces: forces from sources other than the participants in the scenario. These are specified by the user directly, or come from user-specified kinematic constraints.

Inference Methods

AMES builds dynamics descriptions in two phases. The first phase enumerates the forces that the current state contains. The second phase produces information on their values. This subsection discusses the first phase. The following subsection describes the second.

In AMES's limited universe, it is very simple to deduce the existence of forces. External forces come from two sources. The easiest to identify are those that the user specifies directly. For example, a scenario might have a block pushed by some force that arises from interactions outside the system: the user must therefore explicitly tell AMES about the force's existence. The second class of external forces that AMES detects come from motion restrictions imposed by outside sources. These motion restrictions must have reaction forces that provide the necessary constraints.

The existence of gravitational forces are just as easy to infer: the dynamics module deduces a gravitational force for every unique pairing of a rigid body with a gravitational field. Lastly, AMES postulates an action-reaction pair of contact normal forces for every contact in the current state.

Model Contribution

AMES adds equations to the mathematical model that describe each force present in the current state. For user-specified forces, the system simply asserts that the force has its user-provided value. For forces that enforce user-specified motion constraints, no equations are necessary: Newton's second law constrains them to have whatever values necessary to enforce the motion constraints they support. For gravitational forces, AMES asserts that the force value is the value of the gravitational field, scaled by the mass of the object on which each force acts.

Only contact normal forces make a somewhat complex contribution to the mathematical model. AMES asserts that each contact force has no component tangential to the contact locus that generates it. In other words, each contact force must have a direction that is normal to its corresponding contact: hence their description as "normal" forces.

This definition intentionally leaves undefined the magnitudes of the contact forces. The reason is that normal forces are compensating forces: they adopt the minimum magnitude necessary to prevent rigid bodies from penetrating each other. Ensuring that the bodies move tangentially along the contact locus represents this "minimal" effort. Therefore, the kinematic constraints determine the normal force magnitudes.

Dependencies

AMES assumes that gravitational fields and external forces are permanent. Therefore, initial deductions need never be changed. Since their semantics are so simple, those deductions do not depend on any special conditions for validity.

Normal forces change with states' contact configurations; however, as long as a particular set of contacts persists, the above deductions surrounding normal forces remain valid. Note the formulation of normal forces appears to leave open the possibility of attractive normal forces, when textbook-style knowledge states that normal forces can only be repulsive. To understand why this is not a problem, recall that when normal forces become attractive, the contact module understands that this is a sign that the contact is breaking. The justification is that normal forces would only be attractive if the bodies had some tendency to move apart. Therefore, qualitative states always end before normal forces become attractive.

2.5.5 Newtonian Mechanics

Scope

The Newtonian mechanics module is responsible for relating quantities in the state by applying Newton's laws of motion wherever possible. Note that although there are three laws of motion, the first law is merely a special case of the second law. The first law states that objects have uniform velocity unless acted upon by external unbalanced forces. The second law states the mass of a body multiplied by its acceleration, measured from an inertial reference frame, is equal to the sum of the incident forces on that body. When the sum of the incident forces is zero, the second law is identical to the first law. The Newtonian analysis module therefore only records instances of the second and third laws.

In addition to applying these constraints, this module also generates the differential equations that represent the derivative relationships between acceleration, velocity, and position.

Inference Methods

The Newtonian mechanics module generates all possible instances of Newton's second law by essentially performing a free body analysis of every rigid body. This involves retrieving all the incident forces on each body from the dynamics module.

Similarly, it constructs every possible instance of Newton's 3rd law by retrieving from the dynamics module every action-reaction force pair. Action-reaction pairs can be identified by their mirrored source-target relationship, and similarities in force type and point of interaction.

Model Contribution

The Newtonian analysis module generates very straightforward textbook-style equations for each law instance. For Newton's second law, it asserts that the sum of the incident forces equals the mass of a body multiplied by its acceleration. For Newton's third law, it asserts that the values of action and reaction forces are vector negations of each other.

There is a subtlety worth noting about the process, however. AMES states each law in a canonical style: it describes all quantities with respect to the fixed inertial reference frame, using cartesian coordinates. This simplifies the instantiation process, and relies on the conversion modules to relate the results of the law's constraints to variables representing differently measured versions of the same physical attributes.

As previously mentioned, this analysis module also contributes equations that describe the derivative relationships between position, velocity, and acceleration. Note that neither Newton's laws nor the relationships among motion attributes introduce new restrictions on the validity of the model of the qualitative state.

2.5.6 Reference Frame Semantics

Scope

This analysis module defines the semantics of reference frames by providing knowledge about how to convert quantities between reference frames. AMES allows reference frames to be attached to any rigid body in a scenario. The ability to measure quantities in different reference frames allows compact representations of many aspects of the behavior of mechanics systems. For example, contact properties are very easy to describe in terms of relative positions. Relative positions, in turn, can be expressed cleanly in terms of one object's position in another's object's reference frame.

Inference Methods

During the simulation, the task of the reference frame conversion module is to add equations to states' mathematical models that relate variables that represent measurements of the same quantities in different reference frames. Although there are many ways to accomplish this task, AMES operates by relating all attributes in the mathematical model to their values measured in a common reference frame: the fixed frame.

Model Contribution

AMES measures only attributes that describe motion against reference frames. For each of position, velocity, and acceleration, the conversion equations have the same format. The attribute's value in the standard frame of reference is the sum of its value in the non-standard reference frame, plus the non-standard reference frame's value in that kind of attribute, measured against the fixed frame. If the reference frame's value in the attribute is not measured with respect to the fixed frame, it can be obtained by applying the same conversion process. Naturally the conversions depend on there being some sequence of intermediate reference frame relationships that eventually terminates with the fixed frame.

2.5.7 Coordinate Systems Semantics

Scope

The coordinate system analysis module contains knowledge about coordinate system semantics in its ability to mathematically relate measurements of quantities from different coordinate systems, but identical reference frames.

AMES supports two types of coordinate systems. For two-dimensional spaces, AMES uses traditional cartesian coordinates. It describes all cartesian coordinate systems by an offset and a rotation relative to a distinguished coordinate system associated with each reference frame.

AMES also supports specialized parametric coordinate systems that I term *path coordinates*. Path coordinates simplify problems that involve determining the behavior of bodies that are constrained to move along one-dimensional trajectories. Under path coordinates, attributes of a system are measured with respect to unit vectors that are tangential and normal to the trajectory at the location of the constrained body. The position of the body, however, is measured in terms of distance along the path.

This system of measurement simplifies calculations since it separates the influences on a body into components that cause it to translate along its trajectory, and components that cause the curvature of the trajectory. This arrangement also simplifies the task of reasoning about the effects of compensating forces: those forces that constrain the body under observation to its designated path.

Inference Methods

The coordinate system analysis module provides the mathematical relationships between quantities measured in different coordinate systems in much the same way as the reference frame conversion module. It works by generating equations that relate all measurements to their analogs measured against their reference frames' standard cartesian coordinate systems. Combined with the reference frame conversions, the process allows all different measurements of identical quantities to be mathematically related.

Again, similarly to reference frame conversions, AMES applies coordinate system conversions to every quantity in the mathematical model that has been expressed in a non-standard format. The next subsection discusses the structure of the conversion equations.

Model Contribution

Conversions of both cartesian and path coordinates occur in much the same way. The major difference between the two lies in the fact that the rotation and offset of

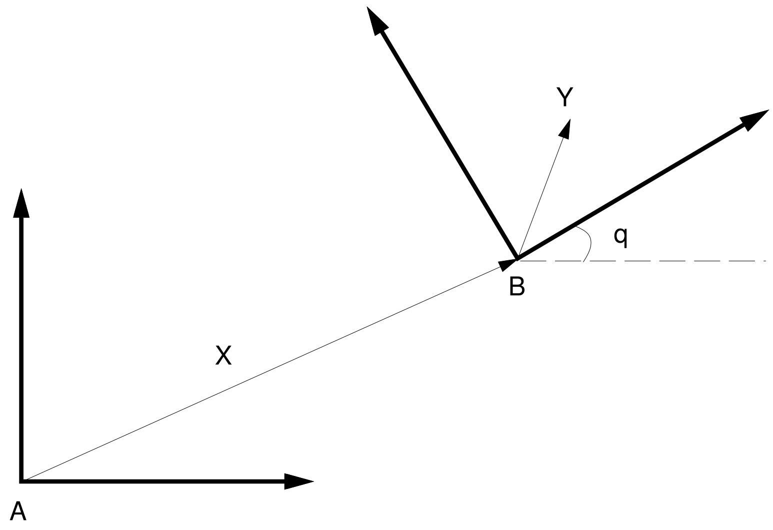

Figure 2-4: Coordinate System Conversion

the cartesian coordinate systems in AMES are constant, whereas in path coordinates they can be variable. The only other difference lies in each system's methods for measuring displacement.

For both types of coordinate systems, the conversion equations have much the same format. Displacement requires somewhat special treatment, however. The diagram below illustrates the general case.

For attributes other than displacement, AMES converts a quantity that has value Y in coordinate frame B by simply rotating Y by the rotation of coordinate system B, angle q. AMES must also add the offset vector X when converting displacements.

For cartesian coordinates, AMES reads the offset and angle information from the description of the coordinate system. For path coordinates, this information comes from geometric operators that, given the parameterized position of an observing body along a trajectory, return the path angle and cartesian position.

2.5.8 Subsequent State Generation

When the mathematical model of a physical system in its current qualitative state is complete, AMES determines when the state terminates, and generates a description of the subsequent state. During the model construction process, the system accumulates a set of preconditions for the model's accuracy. Since states in AMES, by definition, persist only as long as the corresponding mathematical model remains valid, the current state terminates when the first model validity precondition fails.

To find this time, AMES simply attempts to solve for the time when each precondition fails. If there are no solutions, then the current state persists indefinitely, and the simulation terminates. Otherwise, AMES sorts the failing preconditions by time and considers the set of preconditions that fails first. A mathematical subtlety is that only those solutions that have times after the start of the state are valid, since the model itself is not valid before that time.

With the state termination time, AMES solves for the values of the positions and velocities of all the rigid bodies in the scenario. This information forms the basis for the initial conditions of the next state. Positions and velocities completely describe the configuration of the system, and have the property that they are continuous; therefore, their values do not change across the state boundary. This makes these attributes adequate for describing system configurations.

In addition to the values of these attributes, the state termination analysis provides key information about the qualitative properties of the subsequent state, based on the manner in which the previous state ends (i.e., the model precondition or preconditions that failed). This is necessary to the simulation process since the period of state transition is always at the boundary between different behaviors, and the information necessary to disambiguate them is not always contained in the positions and velocities alone.

For example, consider the following scenario, where a particle is at rest touching the underside of a horizontal plane.

For this case, AMES creates an initial state during which the two bodies touch. As should be clear to the reader, this state terminates immediately; therefore, the initial

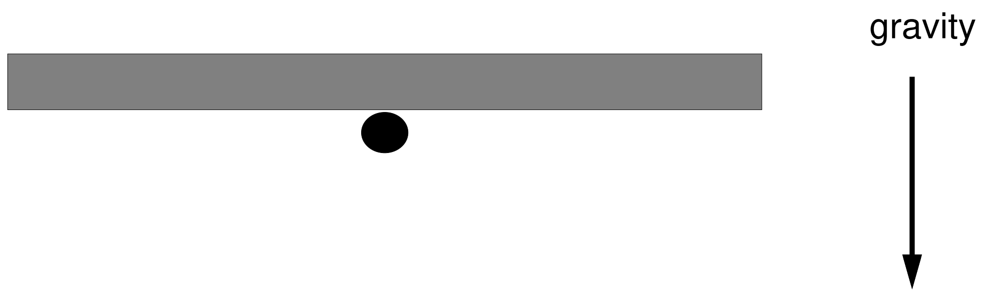

Figure 2-5: Example of State Change Ambiguity

conditions for the subsequent state are identical to the original initial conditions. Additional information is evidently necessary for the next state's analysis: in particular, the system needs to communicate information about how the previous state ended. In this case, we can exploit the knowledge that the initial state terminates because the normal force from the plane to the particle would have to be attractive in order to preserve contact. In other words, there is an applied force on the particle that draws the bodies apart. Therefore, in the simulation's second qualitative state, AMES can correctly assume a free fall situation.

This example suggests that position and velocity information are not sufficient to generate accurate descriptions of states at times of state transition. On the other hand, no more attributes of the subsequent states can be predicted by the previous state's analysis, since all other time varying attributes can change discontinuously across state boundaries. It is therefore important to employ facts about how state transitions arrive in order to describe new qualitative states.

As previously mentioned, it happens to be the case that for the range of problems that AMES addresses, only assertions about contacts generate state termination conditions. Contacts in AMES are binary in nature: they are either present or not present. After a state terminates, therefore, it suffices to simply reverse the status of the contacts associated with the conditions that triggered the state change. We can assume that other contacts remain intact since their termination conditions were not met, and contact depends on position, which is a continuous attribute.

Model Component: *Contact*

Arguments
Rigid body: $Body1$.
Rigid body: $Body2$.
Simple contact locus between $Body1$ and $Body2$: $Contact$

Activation Conditions:
position($Body1$, : wrt $Body2$) $\in Contact$

Deactivation Conditions:
magnitude($NormalForce$) < 0

Qualitative Assertions:
There exists a force $NormalForce$ from $Body1$ to $Body2$.

Mathematical Assertions:
direction($NormalForce$) = angle($Contact$, : at position($Body1$, : wrt $Body2$)) $+ \frac{\pi}{2}$
position($Body1$, : wrt $Body2$)) $\in Contact$

Figure 2-6: Contact Interaction Model Component

2.5.9 Summary

Figures 2.5.9 through 2.5.9 summarize AMES' high level knowledge about the mechanics domain. They organize this information according to a representation scheme called *model components* that the next chapter will describe in detail. The model component representation arose from an analysis of AMES' reasoning paradigm and crystallization of its physical reasoning knowledge.

2.6 Examples

This section demonstrates the methods that AMES uses to analyze physical systems by discussing the program's ability to solve three sample problems. The examples illustrate the range of complexity that AMES can accommodate. The discussion surrounding each problem highlights the elements of the program's approach that provide its power.

Model Component: *Terrestrial Gravitation*

Arguments:
Rigid body: $Body$.
Gravitation field: $Field$.

Activation Conditions: always

Deactivation Conditions: never

Qualitative Assertions:
There exists a force $GravForce$ from $Field$ to $Body$.

Mathematical Assertions:
$GravForce = \text{mass}(Body) \cdot \text{strength}(Field)$

Figure 2-7: Terrestrial Gravitation Model Component

Model Component: *Newton's 2nd Law*

Arguments:
Rigid body: $Body$.
Forces on $Body$: $Forces$.

Activation Conditions: always

Deactivation Conditions: never

Qualitative Assertions: none

Mathematical Assertions
$\Sigma Forces = \text{Mass}(Body) \cdot \text{Acceleration}(Body, : \text{wrt } FixedFrame)$

Figure 2-8: Newton's Second Law Model Component

Model Component: *Newton's 3rd Law*

Arguments:
Force of type $Type$ from $Body1$ to $Body2$: $Force1$.
Force of type $Type$ from $Body2$ to $Body1$: $Force2$.

Activation Conditions: always

Deactivation Conditions: never

Qualitative Assertions: none

Mathematical Assertions:
$Force1 = -Force2$

Figure 2-9: Newton's Third Law Model Component

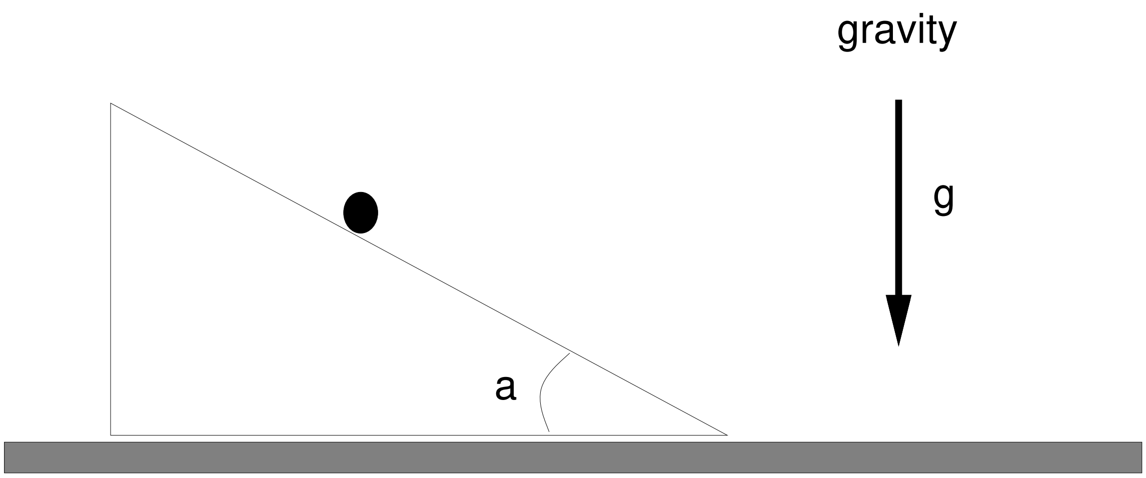

Figure 2-10: Particle on a Wedge

2.6.1 Particle on a Wedge

Consider the scenario depicted in figure 2.6.1. Assume that all bodies begin at rest, and that all contacts are frictionless. The ground's position is fixed, but the particle and wedge are free to move. Terrestrial gravity is present.

There are a number of complexities in this problem that make it interesting to examine:

- Solving for the normal force between the particle and the wedge is difficult, since both are non-inertial reference frames.
- The acceleration of the wedge depends on the magnitude of the normal force from the particle: a circular dependence.
- It is not entirely obvious whether it is possible for the wedge to slip out from underneath the particle and break contact.

As this section will discuss, however, application of AMES' methodical approach generates sufficient mathematical constraints to solve for the unknown forces and accelerations. This then permits the system to completely characterize all motions in the scenario. Together, these elements constitute a complete model of the physical system in its initial qualitative state.

Given the initial configuration of the scenario, AMES' first step is to identify all the contacts. It finds the particle contacts the top of the wedge, and that the bottom of the wedge contacts the ground. Furthermore, there are no initial velocities that would immediately break contact. This contact information allows the system to deduce a number of important facts.

In terms of the scenario's kinematics, the contact information allows AMES to conclude that during the initial qualitative state:

- The velocity of the particle remains tangential to the top of the wedge.
- The velocity of the wedge remains tangential to the top of the ground.

Since the respective surfaces are straight, the derivative relationship between velocity and acceleration implies that the two bodies' accelerations are also constrained to be tangential to their respective contacts.

In addition to these kinematic constraints, the contact information allows AMES to conclude the existence of normal force pairs between the wedge and the particle, and between the wedge and the ground. Also, AMES' domain knowledge constrains the directions of these forces to be perpendicular to the plane of their respective contacts. Note, however, that the normal force magnitudes cannot be directly determined at this time.

Adding to the normal forces, AMES analysis of the scenario's dynamics concludes that gravitational forces influence all three rigid bodies. AMES can determine both the magnitude and the direction of each of these forces since it has information about the bodies' masses and the gravitational field's strength.

At this point, all the high level interactions in the system have been expressed in terms of the kinematics and dynamics of each individual participant. Therefore, AMES is in a position to apply Newton's laws to the situation. Newton's third law equates the magnitudes of the members of each normal force pair. Newton's second law provides free body analysis of each object.

AMES performs free body analysis with respect to the fixed frame of reference to avoid the complications of reasoning about the fictitious forces that non-inertial

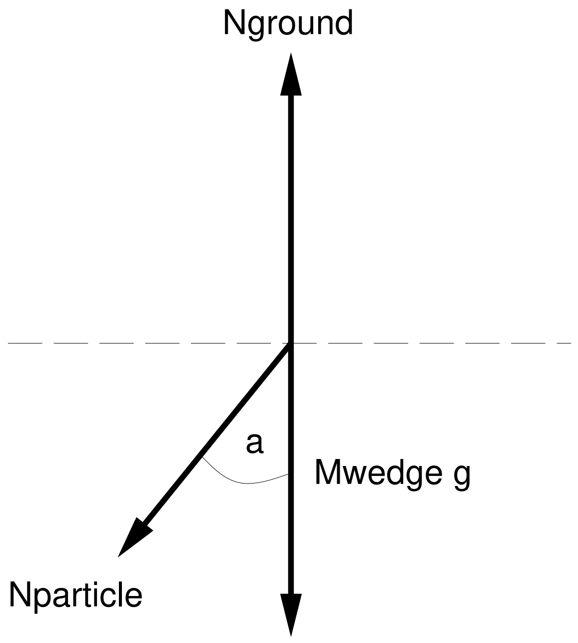

Figure 2-11: Wedge Free Body Diagram

reference frames require. Figure 2.6.1 shows the free body diagram for the wedge. Since the acceleration of the wedge is purely horizontal, the free body diagram provides enough information to determine that the wedge's acceleration has strength $\frac{N_{particle} \sin a}{M_{wedge}}$ toward the left.

The free body diagram for the particle is slightly more complex since the direction of its acceleration in the fixed frame of reference is not entirely clear. Nevertheless, AMES has enough information to solve for this information. Figure 2.6.1 illustrates the free body diagram for the particle.

While it may not seem that AMES has enough information to solve for the normal force $N_{particle}$, it actually can. The acceleration of the particle in the fixed frame of reference $A_{particle \leftarrow fixed} = A_{particle \leftarrow wedge} + A_{wedge \leftarrow fixed}$. This comes from AMES' reference frame conversion knowledge. It is useful since we know that $A_{particle \leftarrow wedge}$ is parallel to the top of the wedge, and we know that $A_{wedge \leftarrow fixed}$ has magnitude $\frac{N_{particle} \sin a}{M_{wedge}}$ and is directed leftward.

Therefore, in the vertical direction, AMES has:

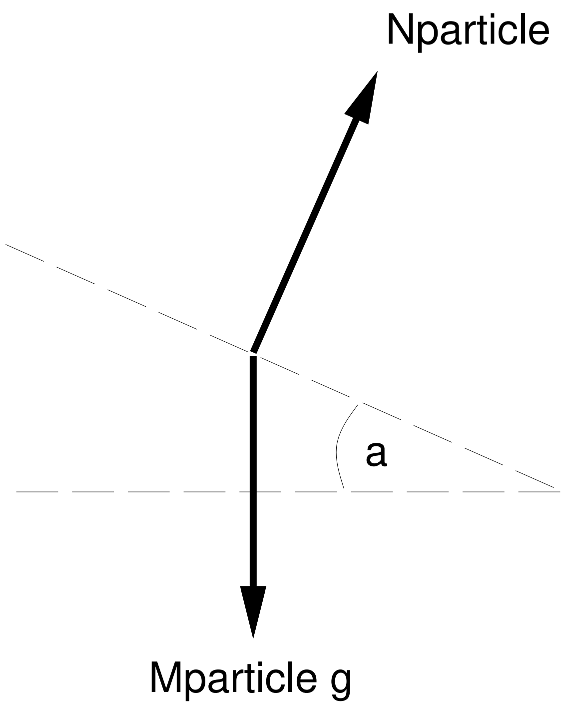

Figure 2-12: Particle Free Body Diagram

$$\begin{aligned} N_{particle}\cos a - M_{particle}g &= -M_{particle}A_{particle\leftarrow wedge}\sin a \\ A_{particle\leftarrow wedge} &= \frac{M_{particle}g - N_{particle}\cos a}{M_{particle}\sin a} \end{aligned}$$

In the vertical direction, by substituting in the above, AMES has:

$$\begin{aligned} N_{particle}\sin a &= M_{particle}\left(A_{particle\leftarrow wedge}\cos a - \frac{N\sin a}{M_{wedge}}\right) \\ N_{particle}\left(1 + \frac{M_{particle}}{M_{wedge}}\tan^2 a\right) &= M_{particle}g\frac{\cos a}{\sin^2 a} \\ N_{particle} &= \frac{M_{particle}M_{wedge}g\cos a}{M_{wedge} + M_{particle}\sin^2 a} \end{aligned}$$

Having solved for $N_{particle}$, AMES can obtain each body's acceleration, velocity, and position versus time. Note that because $N_{particle} > 0$ for all time, the particle does not slip off the top of the wedge. The state therefore ends when either the particle reaches the ground or the wedge falls off its edge.

As mentioned earlier, several features make this example both challenging and interesting: non-inertial reference frames, possible changes in contact, and the complex interaction between particle and wedge. Four key features of AMES' knowledge and inference methods were critical to its performance on this problem.

First, AMES' use of algebraic reasoning was able to capture the complex interaction between the particle and the wedge in the form of a system of equations. In addition to being an appropriate representation, it allows AMES to takes advantage of the success that the field of computer algebra has experienced.

The second important feature was AMES' ability to convert between attributes measured in different reference frames. This allowed Newton's second law to be expressed simply in an inertial reference frame, while also taking advantage of information about accelerations in non-inertial reference frames.

Third, AMES' method of pairing each normal force with constraints on motion during contact guaranteed that enough information was present to solve for all the normal force magnitudes. With complete force information, AMES could then solve for all the motions of bodies and obtain a complete model of the scenario.

Lastly, AMES' knowledge of state termination conditions determined that the wedge would not slip from under the particle, ensuring that the model of the state was valid until the particle reached the ground. Though these four features proved especially useful in this example, they form the basis of AMES' reasoning in all problems.

2.6.2 Two Blocks in a Corner

This second example emphasizes AMES' ability to uncover and exploit multiple restrictions on bodies' degrees of freedom. It addresses the problem of determining the motions of blocks A and B in the scenario depicted in figure 2.6.2.

Assuming the presence of terrestrial gravitation, and external force F, AMES easily solves for the motion of the blocks from rest. The key to the program's performance on this problem lies in its ability to accumulate motion constraints from each

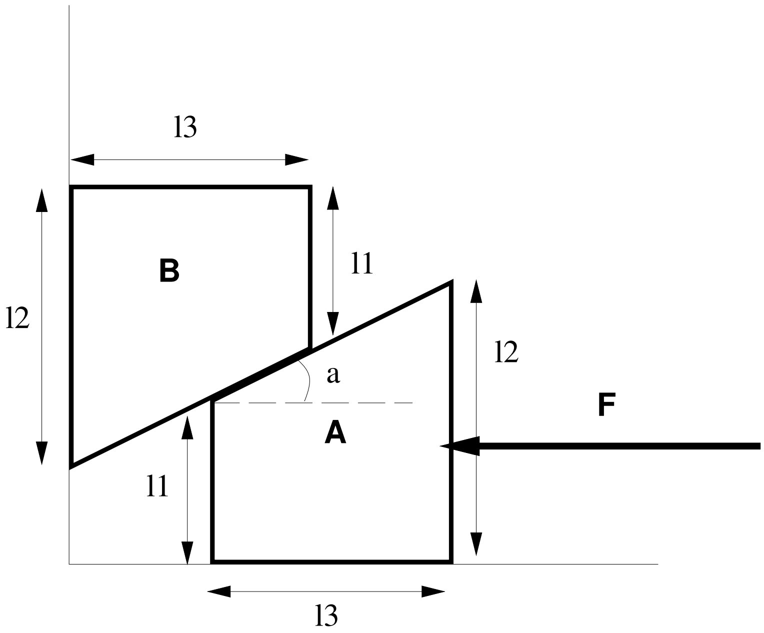

Figure 2-13: Corner Problem

contact and relate them using reference frame conversions.

The contact configuration allows AMES to deduce that in the current state, B moves vertically, A moves horizontally, and that B translates along the top of A. Therefore, AMES generates equations equivalent to the following:

$$\begin{aligned} D_A &= (x_A, 0) \\ D_B &= (0, y_B) \\ D_B &= D_A + (0, l_1) + a(l_3, l_2 - l_1) \end{aligned}$$

With some manipulation,

$$\begin{aligned} (0, y_B) &= (x_A, 0) + (0, l_1) + a(l_3, l_2 - l_1) \\ x_A &= -a l_3 \\ Y_B &= l_1 + a(l_2 - l_1) \end{aligned}$$

$$= \quad l_1 - \frac{x_A}{l_3}(l_2 - l_1)$$

Therefore, the mathematical model that AMES generates completely constrains the relative positions of blocks A and B during the state. This constrains their accelerations and allows AMES to solve for their values with the help of Newton's second law. If N is the normal force between blocks, the law gives:

$$\begin{aligned} m_B a_{By} &= N \cos a - m_B g \\ m_A a_{Ax} &= F - N \sin a \end{aligned}$$

Given the ability to express a_{Ax} in terms of a_{By}, there are enough equations to completely solve the above system of equations.

This example illustrates the manner in which AMES is able to find the net effect of restrictions on bodies' degrees of freedom by accumulating expressions describing those constraints in its mathematical model of the scenario. These restrictions prove necessary for solving for the behavior of many systems.

2.6.3 Particle on a Curved Surface

Consider the situation depicted in figure 2.6.3, where the particle begins at rest at angle a from the top of the sphere. The sphere is fixed to an inertial frame of reference.

This example demonstrates the generality of AMES' representation of contact behavior: the formulation that the program uses correctly determines the magnitude of the normal force from the curved surface. Informal descriptions of normal forces typically state that they balance the component of applied forces in the direction normal to the plane of contact. This definition proves inadequate to characterize contact with curved surfaces, since it fails to account for the centripetal forces that bend the paths of bodies that slide against such surfaces. In addition, this example illustrates the usefulness of *path coordinate systems*, and the normal force test for loss of contact.

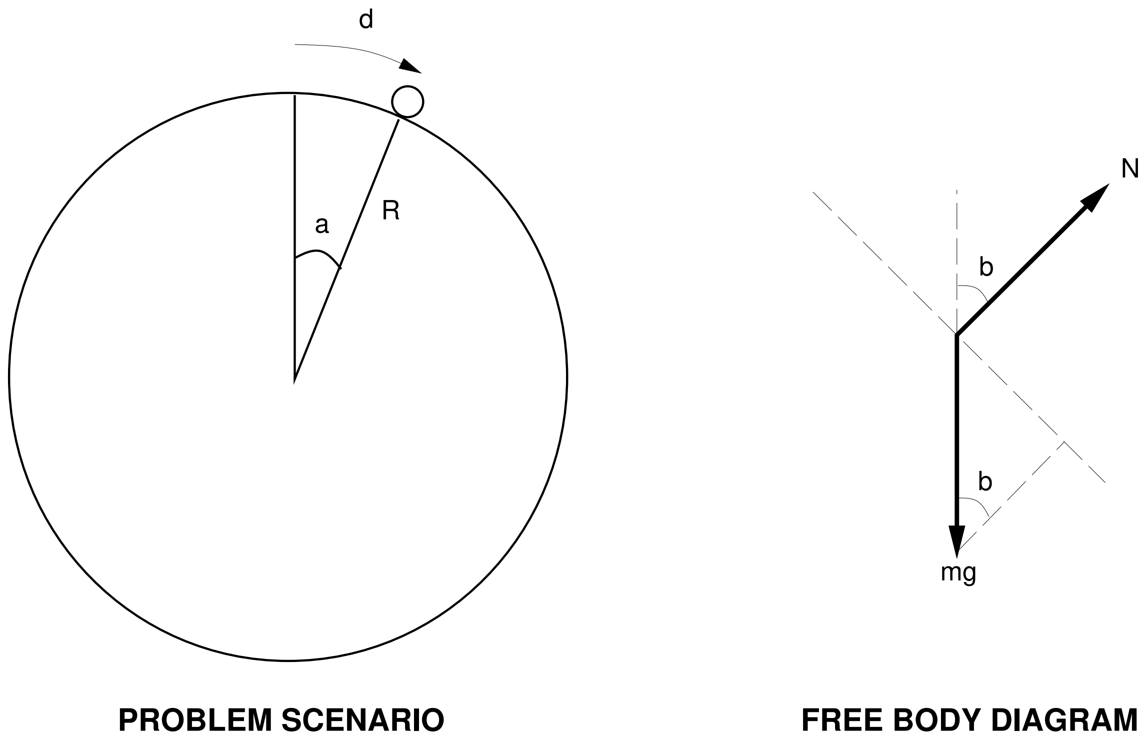

Figure 2-14: Particle on a Sphere

From the contacts in the initial configuration, AMES' analysis can construct the free body diagram depicted in figure 2.6.3. The second key ingredient in the analysis is that the motion of the particle in the current state is limited to the edge of the circle; AMES describes this elegantly by setting up a path coordinate system on the perimeter of the circle, and asserting that the particle's velocity in the normal direction is zero. Combined with the free body diagram, this is enough information to solve for the motion of the particle.

Using knowledge that acceleration is the time derivative of velocity, AMES can deduce that the acceleration of the particle in path coordinates must have a normal component that is $\frac{v_{tangential}^2}{R}$ where in general, R would be the radius of curvature of the trajectory at the position in question.

From this information and Newton's second law, it is possible to deduce that in path coordinates, when the particle is at position d:

$$
\begin{aligned}
a_{tangential} &= g \sin\left(\frac{d}{R}\right) \\
ma_{normal} = \frac{mv_{tangential}^2}{R} &= mg \cos\left(\frac{d}{R}\right) - N
\end{aligned}
$$

The particle therefore contacts the sphere as long as $mg \cos\left(\frac{d}{R}\right) - \frac{mv_{tangential}^2}{R} > 0$.

One important note about this analysis is that while AMES can generate the differential equations that model this scenario, the method assumes that the mathematical engine can solve those differential equations to find out when the state ends and determine the initial configuration of the subsequent qualitative state. In this case, the differential equation is non-linear. Often, when students are asked to reason about this kind of situation, the problems that educators pose do not deal with exact times. Therefore, energy methods become applicable. AMES, however, has no model of when it is possible to omit unneeded information from its analysis to simplify the reasoning task.

Chapter 3

Extensions to AMES

3.1 Overview

Although AMES can only reason about a very limited subset of mechanics problems, many of the principles behind its design support a broader range of capabilities. The current chapter addresses this claim by selecting features from AMES, extracting general lessons from their design, and suggesting how to extend these mechanisms to support a wider range of reasoning abilities.

This discussion occurs in two parts. The first section discusses AMES' basic reasoning paradigm: it looks at issues in knowledge representation and inference methods. The second section builds on this by examining the task of capturing and organizing knowledge about the mechanics domain.

3.2 Representation and Reasoning

3.2.1 Introduction

This section has two objectives. The first goal is to characterize AMES' reasoning methods in a general enough fashion to show how those techniques can accommodate extensions to the program's domain knowledge, or even application of similar methods to other domains. This discussion will also highlight the different kinds of knowledge that such extensions require, and provide a better understanding of the capabilities such reasoning techniques can provide.

Building on this examination of the foundations of AMES' reasoning power, the second goal of this section will be to suggest some ways in which one might adapt this style of reasoning to accommodate a wider variety of problem models.

3.2.2 Knowledge Representation

Overview

This section examines the different classes of knowledge in AMES, and discusses both their role in the reasoning process, and various issues in their representation. A program like AMES manipulates two basic types of information: descriptions of physical scenarios, and general knowledge about the behavior of their domain.

The remainder of this section uses experience with AMES as a starting point for discussing useful representation schemes for these two types of knowledge. This is important for understanding how one might extend AMES' domain knowledge, since the representations provide a concise description of the kind of information required to support changes to the program's behavior.

Scenario Representations

In order to reason about physical systems, programs like AMES must be able to manipulate descriptions of them. This section examines some of the issues raised during the design of AMES' representation for physical scenarios, and discusses the types of knowledge required to support the methods that it uses.

Like many knowledge-based programs, AMES describes the systems it reasons about in terms of objects they contain, and the values of various attributes. Two features make attributes in AMES somewhat unique, however. First, AMES treats any measurable property of a physical system as an attribute. Therefore, attributes need not be limited to describing just individual objects: they may also describe interactions, such as forces. Other examples of attributes in this style could include the velocity of the center of mass of a collection of bodies, or the work one body does on another over a period of time.

The second distinguishing feature of AMES' treatment of attributes is that the program explicitly supports the fact that attributes can be measured in different ways. For example, velocities can be measured with respect to different choices of reference frame and coordinate system. Furthermore, when one adds rotational degrees of freedom, an additional factor is the point on an object that a velocity vector specifically describes.

While it is possible to eliminate this complication by adopting conventions for attribute measurements, the additional expressive power offers several benefits. One advantage is that it allows more compact and lucid expressions of domain knowledge. For example, AMES supports both a definition of contact in terms of relative position, and a definition of Newton's second law in terms of inertial reference frames. Another benefit of allowing attributes to be measured in different ways is that it permits the system to communicate with users in a more intelligible manner: permitting different ways to express mathematically equivalent answers can highlight different regularities in systems' behaviors.

The price of this added expressiveness is that programs that permit multiple descriptions of the same attribute must understand the relationship between the different formats they support. In particular, algebraically-oriented systems such as AMES must know the mathematical relationships between different measurements of the same attribute, since they are an essential part of the quantitative models that describe systems' behavior. Examples of this knowledge in AMES include not only reference frame and coordinate system conversions, as mentioned previously, but also the relationships between position, velocity, and acceleration.

To summarize, designing a scenario description language for a program like AMES involves considering representational issues that include the following:

- The classes of objects in the domain

- The kinds of attributes that physical systems can have

- The types of combinations of objects that each attribute can describe.
- The various ways to measure each attribute
- The relationships between different measurements of the same attribute.

Domain Knowledge Representation

This section characterizes the kind of knowledge AMES requires to predict the behavior of mechanics systems, and develops a representation scheme called *model components* that makes both the form of this information and its recommended usage explicit.

As outlined in the previous chapter, AMES predicts the behavior of physical systems by constructing mathematical models that describe their evolution. A single system often requires different models to describe its behavior during different intervals of time, however. One of AMES' chief tasks, therefore, is to reason about the limits of models' validity, and construct new ones when necessary.

A more detailed perspective on this process comes from considering the configuration space of a physical system: a hypothetical space with dimensions corresponding to each attribute that helps to uniquely describe the system. From this viewpoint, the configuration of a scenario at any single time corresponds to a point in its configuration space. A simulation history, in this scheme, is the path a system traces in its configuration space as it evolves.

Mathematical models, based on knowledge of behavior in the domain, predict the shapes and speeds of such trajectories. A limitation of these models, however, is that typically no single one can predict system evolution in all parts of the configuration space. Therefore, a configuration space might have several regions, corresponding to the ranges over which different models hold. From a theoretical perspective, AMES' reasoning process therefore encompasses the following four tasks:

1. Constructing the predictive model for the region of configuration space that a system inhabits.

2. Finding the boundaries of that region

3. Solving for when the system will cross one of them.

4. Identifying the region that the system will enter next.

Domain knowledge in AMES addresses all four reasoning steps. Before a program can reason about systems' configuration spaces, however, it is essential for them to know the dimensions of these spaces: the attributes that completely characterize the configurations of physical scenarios. One might also think of these attributes as the *state variables* of systems. Note, however, that the notion of "state" here is different from the qualitative states into which AMES partitions its simulations: the former describe instantaneous configurations, while the latter are regions of configuration space over which different behavioral models describes a system's behavior.

Knowledge of the attributes that form a complete set of state variables serves 3 roles in AMES. First, it specifies the information needed to completely describe the initial conditions of a simulation. Second, and in a similar vein, these are also the attributes a reasoner must solve for to identify the configuration of a physical system at the boundaries between behavioral regions. This information is important since the differential equations that model each qualitative state require a complete set of initial conditions to be fully constrained. Third, the initial state variable values are the only attributes that programs can use to determine the behaviors that a system will exhibit during a qualitative state. The reason for this is that this is the only class of information guaranteed to be available as input for the analysis of a state, and until the model of the system is complete, no other information can be deduced.

To adequately serve in these roles, state variables should satisfy the following criteria:

- *Completeness:* The set of state variables must provide enough information to predict all future system behavior.

- *Minimality:* The state variables ideally should not contain redundant information. The most useful systems require the least amount of input: they can accommodate a greater amount of missing information.

- *Continuity:* State variables cannot change discontinuously. This ensures that trajectories in configuration space are continuous, which in turn guarantees that the final configuration of a qualitative state can serve as the initial configuration of its successor. This deduction step is a fundamental part of AMES' reasoning.

In AMES, for example, the state variables of a system are the positions and velocities of every rigid body. These attributes satisfy all three criteria for the domain of rigid body dynamics with no rotational degrees of freedom.

Knowing the structure of configuration spaces in a domain gives one a starting point for formalizing knowledge about how systems evolve from various configurations. The remainder of this section builds on this by presenting a representation scheme for knowledge about physical behavior, called *model components*. The model component representation is an attempt to capture the regularities in the structure of AMES' knowledge about the mechanics domain. The discussion surrounding this representation should highlight some of the issues one must address to extend AMES' knowledge or apply the same approach to other domains.

The design rationale behind model components comes from two observations about mathematical models of mechanics systems:

- *Models are aggregates:* models of the individual features, relationships, and behaviors of systems compose the model of the system as a whole.

- *The constituent parts of models are regular:* a single class of behavior or relationship may appear many times and in many scenarios.

As the name implies, model components exploit the aggregate property by capturing units of knowledge that correspond to the different classes of basic building blocks for constructing mathematical models of physical systems in the domain of expertise. By allowing different ways to instantiate each model component, the representation takes advantage of the regularity of these standard components.

Since model components are the only source of equations for the mathematical model of a system, the set of model component instantiations is as complete a description of a state as the quantitative model. One can therefore think of the set of model component instantiations as a qualitative level of description, since each model component instance describes a particular high level feature of a scenario, such as the existence of a particular contact. The different qualitative states a system can exhibit, under this scheme, arise from the fact that particular instantiations of model components may only be applicable under specific conditions. These conditions mark the extent of qualitative state regions in scenario configuration space.

While the general idea behind model components is relatively simple, there are a number of pragmatic issues that add to the representation's complexity. The mechanics domain knowledge in AMES, for example requires some additional representational facilities. The list below outlines the parts of a model component representation based on these demands. In addition, figures 2.5.9 to 2.5.9 suggest examples of model components for AMES' mechanics knowledge.

- *Arguments:* Each model component describes some aspect of the behavior and relationship of its arguments. A model component can potentially be instantiated once for each way of matching its arguments to features of the current state.

- *Activation conditions:* This is a set of conditions that describes whether, for a particular set of arguments, it is possible to instantiate the model component. The conditions may involve qualitative assertions produced by other model components, or they may test mathematical relationships involving the state variables that describe the initial configuration of the current state.

- *Deactivation conditions:* This is the set of conditions that describes when an instantiated model component instance is no longer valid. In many cases, these conditions are the inverse of the activation conditions. Explicit deactivation conditions, however, are a useful mechanism. In particular, they support tests of attributes that are not state variables, since deactivation tests are applied to complete scenario models instead of state initial configuration information. This allows, for example, contact model components to use normal forces in deactivation conditions, even though they do not qualify as state variables.

- *Modes:* Each model component instantiation exhibits one of possibly several mutually exclusive modes. For example, a friction model component might have static and sliding modes. Each mode contributes different information to a state's analysis. Modes have the features described below:

  - *Mode entry conditions:* The entry conditions for each mode are exhaustive and mutually exclusive. They determine which mode of a model component is active during the current state. Similar to the component activation conditions, they can test either qualitative or quantitative conditions.

  - *Qualitative assertions:* Each model component may make qualitative assertions about the current state. For example, a contact model component might assert that a pair of normal forces exists.

  - *Quantitative assertions:* These are sets of mathematical constraints to be incorporated into the quantitative model of the state to describe the consequences of the behavior the model component describes.

  - *Mode transition conditions:* These are conditions that indicate whether a system will change modes, and which new mode it will enter.

Most of the structure of model components follows from the discussion thus far. The rationale behind dividing each model component into several modes, however, requires some additional explanation. A simpler alternative organization might eliminate modes by turning each one into a separate model component. Modes, however, let one express information that supports more efficient reasoning about state transitions.

A fundamental difficulty in reasoning about state transitions is that when a system is at the boundary between two qualitative states, the complete set of values for its state variables does not always give enough information to determine which state it will enter. For this reason, zero duration states can appear in simulations: the wrong guess results in immediate termination of the associated qualitative state.

The problem that zero duration states introduce, however, is that since no time passes, they leave the state variables of a system unchanged. Therefore, in order to avoid making the same erroneous choice, a simulator must retain some additional information about transitions aside from the final values of state variables: in particular, simulators must somehow be able to identify the region into which a simulation is heading as it crosses state boundaries.

There are two basic methods for doing this. The mechanism AMES uses represents a compromise between them that the notion of model components with modes supports.

Conceptually, the simplest way to identify the direction that a simulation is evolving as it crosses a state boundary is to record the derivatives of its state variables: the gradient. The gradient indicates the side of a boundary that a system will be on immediately after the transition occurs.

To see why this is inefficient, however, consider a state transition caused when two bodies lose contact. In this case, the analysis of the contact state will conclude that the conditions that ensure contact are no longer valid, and that a state change must occur. Next, we would generate a quantitative description of both the system's final configuration and its evolution direction for input to the next qualitative state's analysis.

That information lets a simulator build a qualitative description of the next state. This analysis, in turn, produces a quantitative model similar to the previous state's model, except that it does not include the effects of the contact that broke. While this process gives the correct result, notice that when we reasoned about the end of the previous state, we already knew that its successor would be missing the contact in question. Furthermore, there was a whole host of behaviors that the change in contact did not affect: these behaviors and their associated model contributions continued unchanged into the subsequent qualitative state. The important observation to make here is that the cause for terminating the first state described what was qualitatively different about the second state: reasoning from state variable values was unnecessary.

Communicating qualitative termination information directly avoids duplicating a great deal of reasoning. The drawback, however, is that while more efficient, it is not always possible to apply such methods. For example, consider a qualitative state that ends when two bodies touch. In such a case, although it is clear that the next state will contain a new contact, it is not obvious whether the contact forces will include static or sliding friction.

This is where modes enter into the representation. Modes have transition conditions that indicate not only when to exit a mode, but also the new mode the corresponding model component instance will exhibit in the next state. This allows a system to express connections between related behaviors. Different model components, on the other hand, describe features for which such transition information is not available. When the instantiation conditions of a model component instance change, they only indicate whether or not the instance appears in the qualitative description of the next state: also a useful way to communicate qualitative information, but not quite as flexible as mode transitions.

3.2.3 Inference Methods

This section describes *algebraic simulation*: an inference method that applies knowledge represented as model components to the task of simulation. Much on this subject has already been presented to explain the design rationale behind the model component representation. The primary objective of this section is therefore to summarize and address a selection of detailed issues.

As mentioned earlier, model components provide the knowledge that allows a simulator to construct a mathematical model of a physical system, identify the limits of its validity, reason about when the system will reach those boundaries, and provide the information necessary for analyzing the system's next qualitative state in the same manner. This section examines the inferences involved in each of these tasks.

Mathematical Model Construction

Constructing a mathematical model of a physical system involves finding all possible ways to instantiate the model components that compose the simulator's domain knowledge. In general, this involves checking, for each possible way to instantiate a model component's arguments, whether the activation condition holds, and determining what mode is appropriate given the system's configuration. Note that for simplicity, I will delay discussing how to use information from the previous state's analysis until later.

Finding all possible ways to instantiate model component arguments is simple exhaustive enumeration. Checking instantiation conditions and mode entry conditions involves looking at quantitative information about the state's initial configuration, and examining qualitative information that other model components generate when they are instantiated. The former class of conditions is easy to check since all required information is available at the start of model construction. The latter class introduces dependencies between model component instantiations, however.

While the symbolic conditions and assertions that model components manipulate might suggest rule chaining methods, experience with AMES suggests that the qualitative assertions from each model component are sufficiently constrained that it is possible to compute a graph of qualitative data dependencies and simply explore instantiations in a topologically sorted order.

Another reason to explore model component instantiations in a methodical fashion is that some reasoning requires considering the additive effects of various influences. For example, reasoning about the effects of forces requires one to sum their effects. Instantiating model components for Newton's second law, therefore, requires a mechanism to ensure all forces have been accounted for before proceeding. Exploring model component instantiations along data dependency paths can guarantee this.

Once we have found all possible instantiations of model components, their individual model contributions collectively form the model of the system in its current qualitative state.

State Termination Analysis

Each instantiated model component provides constraints on the validity of its contribution to the mathematical model of the system. In particular, an activated model component instance is only valid as long as its deactivation conditions and mode transition conditions are false. In addition, each potential model component instance whose instantiation conditions fail in the current qualitative state also provides constraints on the validity of the current model: when their activation conditions become true they add new elements to a system's behavior, rendering the old model inaccurate.

To determine when a qualitative state ends (i.e., when the current model is no longer valid), a simulator must use the system's model to solve for the termination condition or conditions that fail first.

Deductions About the Next State

After determining the conditions that end a qualitative state, an algebraic simulation must generate information to support the analysis of its successor. There are two types of information to provide. First, the simulator must deduce the initial configuration of the physical system's next qualitative state. It does this by solving for the final configuration of the current state. The requirement that state variables be continuous guarantees the correctness of this step.

The second class of information the simulation contributes to the next state's analysis is qualitative in nature. It communicates inferences about the next state based on the manner in which the current state terminates. In particular, with respect to the next state, newly met instantiation conditions imply their corresponding model component instances become active, and vice versa for instantiation conditions that fail. In addition, mode exit conditions imply that the appropriate mode transitions occur in their respective model components instantiations.

In building the model to describe the next state of the simulation, the above qualitative inferences describe the high level ways in which that state differs from its predecessor. The remaining differences between the models of the two states come from the ways in which other model component instantiations depend on these changes. All unaffected model component instances from the previous state, however, remain active in its successor, since there is no reason to believe that they should be excluded. This kind of reasoning about inference dependencies suggests that truth maintenance systems [27] might be an appropriate technology for managing this task.

3.2.4 Reasoning Extensions

Overview

With a solid understanding of the general reasoning paradigm that underlies AMES' abilities to simulate mechanics systems, it becomes interesting to discuss some of the different ways in which these basic methods can be extended to provide additional functionality. This section has two parts. The first part suggests some ways that AMES' simulation reasoning can support additional kinds of problem solving behavior. The second portion of this section looks at some ways to make the simulation process itself more powerful.

Problem Solving Extensions

The problems in a typical mechanics text fall mostly into a few standard categories. As suggested in the introduction, the algebraic simulation paradigm seems to support a large portion of these typical problem types with very little additional machinery. This section briefly discusses how to apply algebraic simulation to the following problem types:

- *Analysis problems:* determining the value of an attribute at some time, or over an interval of time.
- *Parameter selection problems:* determining the value of a parameter that makes a certain outcome occur.
- *Relationship problems:* determining the value of an attribute in terms of some other attribute.

- *Proofs:* showing that some condition holds.

Simulation-style reasoning adapts very easily to all the above problems, since generating a complete description of the behavior of algebraically described systems, as algebraic simulation does, provides a superset of the required information.

For example, analysis problems might begin with an algebraic simulation. Then, obtaining the desired information about the problem scenario would simply require solving for it using the appropriate mathematical model. One source of additional complexity is that many problems specify times relative to events in the scenario: for example, one might want to find the speed of a particle when it slides off an inclined plane.

Such problems merely require a two stage reasoning process. The first stage solves for the relevant time by first looking for the qualitative state where the timing event occurs, and then using the model of that state to solve for an exact time. The second stage uses the same model to solve for the desired attribute at that same time (or interval of time as the case may be).

The basic approach to analysis problems serves equally well for proofs and relationship problems, with minor modification. Relationship problems are the same as analysis problems, except that there are constraints on the form of the answer. Producing the correct form depends on the abilities of a program's underlying algebraic manipulation machinery: simulation methods provide a complete enough set of equations to support such operations.

Proofs ask the reasoner to determine whether a particular hypothesis about a system is true. Hypotheses about the behavior of a system can be solved by simply solving for the characteristics that are the subject of the hypothesis, and comparing the results. Proofs that hypothesize about more abstract subjects than behavior in specific scenarios, however, require reasoning capabilities beyond those explored in this thesis.

Parameter selection problems begin to enter into the territory associated with design, as opposed to analysis: the focus of the current research work. Nonetheless this particular problem type is constrained enough for algebraic simulation to accommodate. The key lies in the ability to represent the parameters in question by symbolic constants, and simulate without deciding the actual value. Recall that AMES' algebraic simulation paradigm was designed specifically to accommodate such abstractions. After running the simulation, parameter selection becomes a simple matter of assuming the desired result and solving for the constraints it imposes on the parameter in question.

Note that there is an important class of complications that can arise in such problems, however. By leaving a parameter completely unconstrained, it may not be possible to completely determine uniquely the way the system will evolve. Such situations require exploring several alternative behaviors in parallel: a feature that the methods presented in this thesis do not support, but can be extended to handle, as the next section will discuss.

Simulation Scope Extensions

The algebraic simulation reasoning paradigm presented earlier in this chapter appears well suited to the task of predicting the behavior of systems with algebraically described parameters in well-defined domains. While this style of reasoning is characteristic of most introductory mechanics, some problems require additional methods. This section outlines various ways that one might expand the reasoning scope of algebraic simulation.

One way to extend the method's usefulness is to augment its algebraic methods with other styles of quantitative reasoning. For example, numerical reasoning might be useful when this level of detail is available, and the equations that model systems cannot be solved analytically. Since the equations that algebraic simulation generates do not depend on algebraic reasoning in any way, they easily support this and any other changes in the way the quantitative portions of the system operate.

In the other direction, it might be useful to extend the quantitative methods to include more abstract reasoning in the style of QSIM [21]: a system that abstracts the values of variables by describing their magnitudes relative to distinguished "landmark" values. This kind of reasoning has generated interest due to its ability to provide a high level of abstraction for reasoning about differential equations. It proves particularly useful for making generalizations, identifying qualitatively different kinds of behaviors, and accommodating incomplete information.

In addition to providing additional mathematical reasoning facilities, research on QSIM and similar approaches have generated a host of techniques for managing ambiguities that arise in simulations where exact values for parameters are unknown: a problem that can arise at the algebraic level of abstraction, as mentioned previously. This problem manifests in the current thesis work whenever it is impossible to predict, from a model of a qualitative state, what condition will terminate that state first. Without a unique answer, it is not possible to disambiguate the next state that a scenario enters.

As is discussed in much greater detail in the QSIM literature, one way to accommodate this problem is to explore all possible outcomes in parallel. Variations on this theme are also possible, depending on the ultimate problem one wishes to solve. The important point here is that the methods developed for AMES generate models and corresponding validity conditions that are purely mathematical and therefore are neutral enough to allow a wide variety of different policies for managing ambiguity.

At a higher level from these mathematical reasoning issues, another way to make algebraic simulations more powerful is allow it to reason about a wider range of scenario descriptions. This section describes one useful type of improvement. The standard model for algebraic simulation takes as its only input the description of the initial configuration of a scenario. The simulator then determines all future behavior from its domain knowledge. This organization precludes reasoning about a whole host of phenomena that have effects in the domain of interest, but whose underlying mechanisms are beyond its scope.

Such situations are typical of problems requiring energy or momentum conservation techniques. For example, consider the problem of finding the final velocity of a wagon after a person throws several bowling balls in the direction opposite the desired direction of travel. In problems such as this one, the net effects of the balls being thrown are typically given, but the actual mechanisms that create the forces involved (e.g., chemical processes in the person's nerves and muscles), are outside the scope of mechanics reasoning.

Nevertheless, we still want to be able to reason about such scenarios without bringing in detailed knowledge about biomechanics and other such external information. The solution to this problem has two components. One component involves describing the effects of events whose causes are outside the domain of expertise. The second component involves reasoning about their effects.

Adding a special model component into a simulator's domain knowledge seems a simple solution for describing the effects of events that are beyond the scope of a program's domain of expertise. Using this scheme, existing machinery determines when special events occur, and updates models to include their effects. Reasoning about these events in such cases then requires no changes.

The real complications enter into the picture when the effects of external events are not specified in complete detail. For example, the wagon example might give enough information to use momentum conservation methods, but omit the details of the forces that the person exerts on each ball. Algebraic simulation, however, assumes perfect information about scenarios, and operates on the assumption that it is possible to solve for any desired quantity. This therefore introduces two new twists to problem solving. One new aspect of reasoning might involve reconstructing a complete picture of a scenario's configuration from user-specified partial information so that reasoning can proceed after an external event occurs. Another new aspect to reasoning might be selectively simulating those aspects of a scenario about which it is possible to reconstruct enough information.

While these remain research issues, the key likely lies in emulating the reasoning people use to accommodate these difficulties. Such problems typically require problem solving techniques that employ conservation laws: explicit methods for abstracting over specific mechanisms to solve for a process' net effects. While it is possible to give AMES formulations of the conservation laws themselves, this is only a small part of the solution: what is crucial here is the style of reasoning that we associate with their application.

3.3 Mechanics Domain Knowledge

With a more general model of the representations and reasoning methods that support AMES' reasoning abilities, it is natural to explore issues in capturing and organizing domain knowledge for use within such a paradigm. This section has two parts. The first uses experience with AMES' design to suggest some general ideas for organizing mechanics domain knowledge. The second part examines through specific examples how these ideas, and the representational machinery described in the previous section might support various extensions to AMES' knowledge-base.

3.3.1 Principles in Organization

The model component representation scheme outlines the type of information that one needs to capture in order to build mathematical models of physical systems; however, left open is the challenge of partitioning knowledge about a domain into such units. This section presents some ideas on this subject, learned from experience with AMES' design.

The knowledge engineering process for AMES, and initial research into extending the program's domain knowledge have produced 3 ideas for organizing mechanics knowledge for use in algebraic simulation:

1. Decompose knowledge into units small enough to be combined to model any physical situation, but large enough so that all the effects of a particular behavior are recorded in the same place.

2. Use human qualitative terminology to guide model component construction: typically high level distinctions translate into different modeling primitives. Analogies between informal concepts and the knowledge in each model component also help make the represented knowledge more intuitive.

3. Layer object behavior knowledge on top of basic point mass mechanics principles.

The first two points give some general guidelines for the level and kind of granularity that is desirable when constructing model component representations of physical domains. As such, the advice remains relevant when dealing with domains other than mechanics. The last suggestion addresses overall knowledge-base organization, and while specific to mechanics, it is based on reasons that may also prove useful beyond the field's immediate scope. The following discussion explores each of the above suggestions in more detail

Level of Granularity

It should be fairly clear why one should decompose knowledge into units small enough to be assembled into models of any physical situations of interest. Note, however, that instead of advocating some specific level of granularity, this guideline suggests that a program's knowledge-base should be tuned to the types of systems it is supposed to reason about.

For example, higher level abstractions and more specialized approaches prove useful in reasoning about the subset of mechanics that deals with machinery. Engineered artifacts like mechanical devices have highly constrained behavior, and are constructed largely from standard parts. Knowledge representations that address typical collections of objects as functional units as opposed to arbitrary configurations of individual parts allow efficient reasoning about problems this subfield.

Although representing small pieces of domain knowledge tends to improve modularity and support generality in a reasoner, there is a lower bound beyond which decomposing knowledge any further becomes counterproductive. Keeping knowledge in as large units as possible without compromising generality has the effect of reducing the amount of reasoning required to build models. More importantly, however, is that larger units of knowledge illustrate the regularities within a domain more clearly. Of course there is a limit to this: units of knowledge can be too large, even though they may be capable of modeling all situations of interest. If model components start to have redundant content, this hints at common behaviors underlying different pieces of knowledge. In such cases, the representation should reflect this structure for both compactness and clarity.

Representing Qualitative Behaviors

When designing AMES, it was extremely useful to examine the terminology of the domain to guide organization of the program's domain knowledge. The reason is simply that, people have a great deal of both formal and intuitive knowledge about physical systems. These often reflect underlying regularities of the domain that form modular units suitable for representation as model components. Furthermore, some human domain knowledge is explicit enough to more or less directly translate into model components. For example, each of Newton's laws translates into a separate piece of knowledge in AMES.

Aside from helping to assemble complete descriptions of knowledge in the domain, representing the concepts and methods that people use allows the knowledge-base to be easier to understand and debug. Nearly all of AMES' domain knowledge came directly from studying standard human reasoning abstractions and making them more explicit. This suggests that the domain has structure that model components can capture in an intuitive fashion.

Representing Object Semantics Using Point Mass Mechanics

Mechanics texts provide very explicit descriptions of mechanics at the free body diagram level: the behavior of isolated point masses. They offer much less concise information about to interpret the interactions between entities such as extended rigid bodies and ropes, however. Part of the reason for this is that intuition supplies much of the necessary information. Unfortunately, it also hides much of the knowledge needed to allow computers to do similar reasoning.

In organizing mechanics domain knowledge, it seems useful to keep knowledge about point mass mechanics separate from knowledge about the behavior of the various extended object classes. The reason is that the two kinds of information are distinct in two important ways. By keeping them separated, one can reap the usual benefits of modular design: ease of both comprehension and maintenance.

The first way in which the two kinds of knowledge are distinct is that they reflect different levels of abstraction: all object interactions in the mechanics domain can be expressed in terms of the kinematics and dynamics of point masses. Furthermore, conventional wisdom in mechanics advocates organizing problem solving in this manner, explicitly reducing all high level behavior to free body diagrams.

The second reason for separating the two bodies of knowledge is that, as mentioned above, point mass mechanics have lent themselves to more explicit description than the behavior of extended objects. Separating the two kinds of knowledge therefore helps contain the portion of the knowledge-base that is likely to require the most modification.

3.3.2 Extending AMES' Domain Knowledge

Based on experience with AMES, this chapter has presented a number of general ideas for designing programs that can predict the behavior of physical systems, especially in the mechanics domain. The goal of this section is to evaluate some of these concepts, and suggest how to extend AMES to reason about a more complete range of mechanics scenarios. Toward this end, this section presents several ideas for building a mechanics knowledge-base that encompasses a broader scope of behaviors than AMES' expertise in two-dimensional frictionless rigid body systems.

Friction

One of the main challenges in representing knowledge about friction is characterizing its compensating nature: it resists relative motion between contacting bodies, but only up to a limit determined by the surfaces' coefficient of friction and the contact normal force. In talking about friction between rigid bodies, people typically make a distinction between static and sliding friction. This distinction is useful to capture in a formal representation since, in each case, friction compensates in a different manner.

Therefore, to extend AMES to reason about rigid body friction, one might use a model component that has the following structure.

- **Arguments**: ?rigid-body-1 ?rigid-body-2
- **Activation Condition**: ?rigid-body-1 contacts ?rigid-body-2
- **Deactivation Condition**: ?rigid-body-1 does not contact ?rigid-body-2
- **Mode 1**: static friction

  Entry condition: relative velocity = 0.

  Qualitative assertions: there is a friction force from ?rigid-body-1 to ?rigid-body-2.

  Model contribution: acceleration of ?rigid-body-2 relative to ?rigid-body-1 in the plane of contact = 0. The direction of the friction force is in the plane of contact.

  Exit condition: friction force $\geq \mu N$ causes transition to sliding friction mode.

- **Mode 2**: sliding friction

  Entry condition: relative velocity > 0.

  Qualitative assertions: there is a friction force from ?rigid-body-1 to ?rigid-body-2.

  Model contribution: friction force magnitude = μN. Friction force angle = opposite direction of velocity of ?rigid-body-2 relative to ?rigid-body-1.

  Exit condition: relative velocity = 0 causes transition to static friction mode.

In addition to a model component like the above, AMES would need attributes that describe the coefficients of friction between various pairs of surfaces. Note that the model above permits different coefficients for static and sliding friction: a factor that many mechanics problems explore.

In studying the above suggestion for a model component representation of friction, note that modes handle the difference in behavior when friction is above and below its magnitude threshold. Another interesting feature is the way the representation describes the magnitude of the static friction force: the constraint on relative acceleration determines its value. This way of expressing the constraint automatically accounts for the effects of "fictitious" inertial forces. Lastly, note that the qualitative assertions about the existence of various forces permits easy reasoning about Newton's laws.

Universal Gravitation

Universal gravitation, in contrast to the terrestrial gravitation model that AMES used, is relatively easy to implement, since the values of these forces are constant and easy to determine. The model component might look something like:

- Arguments: ?body-1 ?body-2

  Qualitative assertions: there is a force from ?body-1 to ?body-2.

  Model contribution: force along line between centers of mass, with magnitude given by $\frac{GM_1M_2}{r^2}$

The key challenge in incorporating universal gravitation into AMES is not so much representing its behavior, but rather in determining when it is a more appropriate model for gravitation than terrestrial gravitation.

Momentum and Energy

AMES currently conducts all its reasoning using forces and the time derivatives of position. Nearly every text on mechanics, however, emphasizes that reasoning in terms of energy or momentum can often be a powerful technique. The first step in allowing AMES to reason about these attributes might be to give AMES knowledge of their mathematical definitions as path and time integrals of force. This is simple enough, however, it helps little, adding only an alternative way to formulate knowledge that AMES already has.

The second step toward leveraging the power of energy and momentum perspectives is to provide knowledge about how these attributes can be measured directly from other system attributes. Toward this end, formulae for gravitational potential energy, kinetic energy, and momentum as the product of mass and velocity would allow AMES to solve certain problems in a simpler fashion. For example, the speed of a rollercoaster or pendulum might be found without resorting to integrating the effects of gravitational forces along a body's trajectory.

Although this type of alternative perspective is useful to people, for a computer, it might sometimes be easier to solve a complicated integral than to search through redundant mathematical descriptions of systems for solutions that have simpler integrals. This is not to say that momentum and energy methods are not useful, however.

For a system like AMES, the real value of these techniques might lie in yet another way that people find them useful: their ability to abstract away the detailed features of certain complex interactions, but at the same time completely summarize their net effects. For example, collisions involve complex interactions that mechanics texts typically abstract by indicating only how momentum and energy are conserved in such interactions. The next section explores reasoning about collisions in greater detail.

Collisions

Reasoning about collisions requires momentum and energy techniques, since the classic model of rigid bodies is ambiguous about the details of the forces that occur during these events. In addition to being capable of reasoning about the basic definitions of these attributes, and having information about the elasticity of colliding bodies, a simulator must recognize collisions as events that it must use special case reasoning to handle.

The reason that collisions require special case reasoning is that traditional mechanics models do not provide enough information to perform a standard simulation of the interactions between colliding bodies. Therefore, like people, programs must reason about these events by treating them as instantaneous events that convert particular input attributes into particular output attributes based on special rules. For collisions, these special rules are momentum and energy conservation.

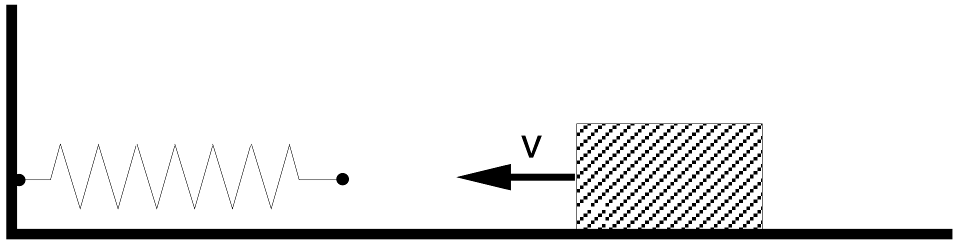

Figure 3-1: Spring Example

While these laws still apply outside of collisions, what is special about collision reasoning is that it ignores other kinds of domain knowledge. Not only is there insufficient information to apply this knowledge (e.g., normal forces are undefined), canonical collisions also involve behavior that violates the basic continuity assumptions of algebraic simulation.

Springs

Springs introduce a new class of objects into AMES' knowledge of mechanics. Their behavior is quite simple, however, and would require very little additional work to represent. The canonical model of springs only deals with how they interact with other bodies at their endpoints. It is therefore possible to model springs as two massless endpoint objects, that have a special force between them that attempts to maintain their separation at the equilibrium spring length.

Using this kind of perspective, knowledge about spring endpoints' contact conditions and normal forces are identical to those for rigid bodies. The only new knowledge required would be a characterization of the spring force. This information comes directly from textbook definitions of Hooke's Law ideal springs: the force they exert is inversely proportional to the displacement of the endpoints from their equilibrium separation. Both the spring constant of proportionality and the equilibrium spring length would be new physical attributes that springs exhibit.

To see how AMES might apply such knowledge, consider the simple problem depicted below.

AMES would conclude that the initial state persists until the block reaches the right endpoint of the spring. At that time, a new contact would appear. AMES' analysis of the system would identify the following constraints:

- Since the state lasts only as long as the contact is present, the contact model component asserts that in the current state:

$$position_{block} = position_{spring\ endpt}$$

- From the spring restoring force model component, the restoring force is:

$$F_{spring} = -k\,(position_{spring\ endpt} - position_{equilibrium})$$

  since the other endpoint of the spring is fixed.

- From the free body diagram model component instance for the spring endpoint, we have:

$$F_{spring} + F_{spring\ to\ block} = 0$$

  since the mass of the spring endpoint is zero.

- From the free body diagram model component instance for the block, we have:

$$F_{block\ to\ spring} = mass_{block} \times acceleration_{block}$$

- Combined with an instance of Newton's third law, we can use the above equations to conclude that the block decelerates in proportion to its proximity to the endpoint of the spring against the wall.

Note that the model of the spring is problematic in several ways. For example, it is difficult to decide what behavior results when an endpoint experiences a force that has a component perpendicular to the axis of the spring. Also, the above model allows behavior such as bodies passing between the endpoints of a spring. It is important to note, however that these are inherent limitations of the canonical spring model that introductory mechanics uses. The key point to understand here is that AMES can experience problems with incomplete models. This means that at a higher level, one must select models that are valid for the types of scenarios one wishes to analyze. Performing this model selection task by computer is an area for future research.

Ropes

Ropes are similar to springs in that they are one-dimensional objects that have interactions at their endpoints. Important distinctions, however, are that there is a length constraint between endpoints instead of a force constraint, and that the model for ropes accounts for interactions that occur along the length of a rope.

The constant length constraint for ropes is similar to the rigidity constraints for rigid bodies in that there is a compensating force to enforce it. In the case of ropes, the compensating force is rope tension, and like contact normal forces, it can be modeled as a force that has whatever magnitude necessary to prevent the rope from stretching. Dividing rope behavior knowledge into two modes can model the fact that they can have tension, but not compression.

There are two remaining issues to address to allow reasoning about ropes. Both involve representing knowledge about the interactions that ropes can have along their lengths. One issue is characterizing the nature of those interactions. Here, the fundamental insight that people use to reason about ropes is that they are massless, and tensions are constant throughout their lengths (for frictionless ropes). The former means that forces on any segment of rope sum to zero, and the latter fact means that tension pulls on both sides of any segment of rope equally.

In terms of representing the effects of ropes on the bodies they contact, this information gives a way to characterize the forces that ropes exert. Consider the reasoning depicted in figure 3.3.2. To reason about the dynamics of the rope's midpoint contact, a program like AMES can focus on the segment of a rope in contact. The forces on the segment of rope must sum to zero; therefore, contact forces must balance the tension forces. This also constrains the force the rope exerts on the body by Newton's third

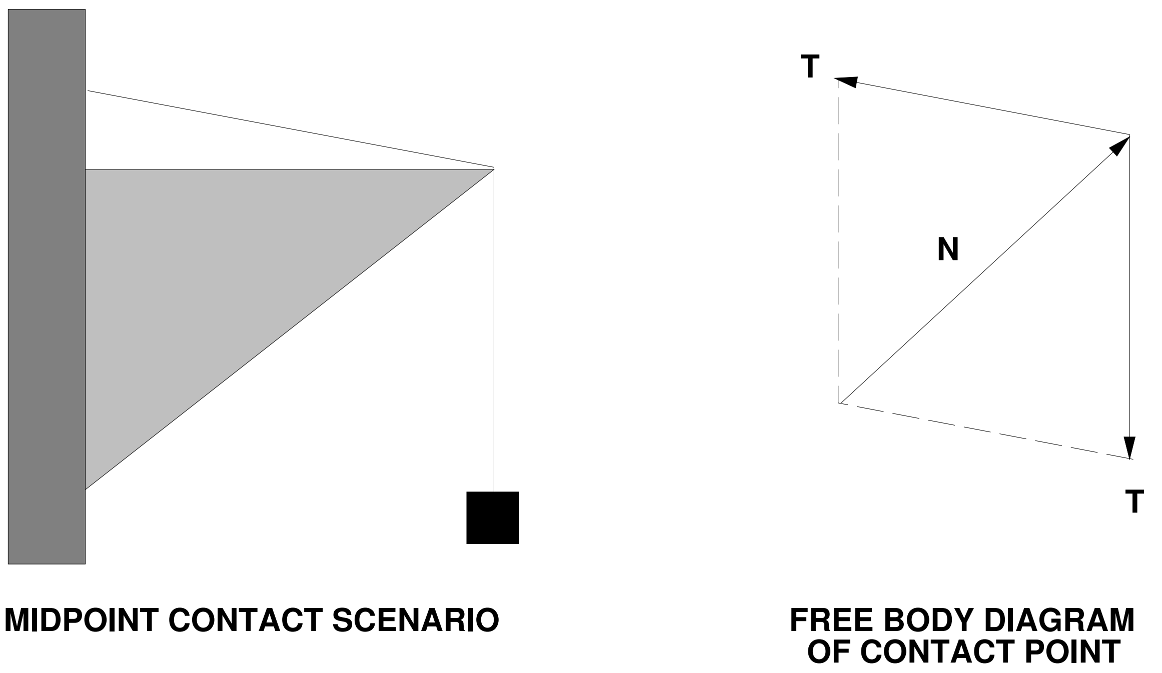

Figure 3-2: Rope Contact Dynamics

law. This kind of relationship should be simple to encode in a model component that has arguments that correspond to contacts between bodies and segments of rope. The only criterion is that the system needs a representation of rope shapes that allows such arguments to be properly instantiated.

That leads into the second issue surrounding the contacts ropes have along their lengths: reasoning about the shape of ropes. Reasoning about ropes under tension is not difficult, since they form convex hulls (or can be decomposed into convex hulls, if wrapped completely around objects). Methods for finding convex hulls are well understood. The difficult part of the problem is that the canonical model of ropes is underconstrained when dealing with loose ropes. The situation is important to reason about, however, since loose configurations determine the way that ropes wind around objects when pulled tight.

A promising idea for handling this problem is to treat ropes as if they had minuscule spring constants and zero equilibrium length when in their loose state. This way, a rope's shape would always form a convex hull, greatly simplifying reasoning and representation. Such a representation permits correct reasoning about how ropes can be woven between objects when loose. Furthermore, since convex hulls have simple shapes, it is very easy mathematically to track during the loose mode, what contacts will occur when the rope becomes tight.

Three-Dimensional Geometry

AMES reasons about two-dimensional scenarios only. Extending the system to handle three dimensions is theoretically a straightforward extension of the underlying mathematics. The additional complexity of reasoning about additional degrees of freedom, however, might raise efficiency issues that such an extension would need to address. A useful method people use is to decompose three dimensional problems into several two or one-dimensional problems. The general task of problem re-representation as a solution method was the subject of [19]; however, the subject still requires a great deal of research.

Rotational Dynamics

Similar to three dimensional geometry, rotational dynamics add additional degrees of freedom that raise the complexity of the domain. Expanding AMES' expertise to handle rotational dynamics begins with basic definitions of the torques, angular motion, and their relationship. In addition, AMES' description language must be extended to distinguish such features as the point of application of forces, and bodies' mass distributions and inertial moments. It is likely, however, that the same style of canonical model assembly that AMES used will prove applicable to a large class of rotational dynamics problems.

Some areas for further research, however, will include:

- For efficiency, recognizing when it is necessary to consider rotational features of scenarios.

- Handling limitations of domain models. For example, it may be difficult to characterize the torque that a normal force provides, since such forces can be distributed across extended surfaces.

Chapter 4

Directions for Future Research

4.1 Overview

The previous chapter explored several ideas for extending the reasoning paradigm developed for the AMES program to a wider range of problem types and domain knowledge. That chapter also highlighted some of the ways in which AMES' problem solving methods were lacking, however. This chapter builds on that discussion by outlining an agenda for future research in three major areas: efficiency, more general reasoning, and modeling.

4.2 Reasoning Efficiency

4.2.1 Problem Overview

AMES modeled qualitative states of scenarios using canonical sets of mathematical constraints between system attributes. Since the equation generation process must guarantee that complete information is present, the equation sets can be on the order of 50-100 equations for a typical mechanics problem. The job of the quantitative reasoning engine is to sort through these and solve for all desired quantities: values of state variables during state transitions, conditions for building models, and conditions that cause change of qualitative state.

Since AMES used the user as its algebraic engine, it is unclear how difficult it is to solve this task by computer. With more complex scenarios and broader domain knowledge, such brute force methods might not be efficient enough to be practical.

4.2.2 Research Agenda

The first step in addressing this problem would be to implement a quantitative reasoning engine for AMES to obtain accurate information on the importance and difficulty of the problem. AMES' heavy dependence on algebraic reasoning was based on the observation that computers have been reasonably successful at such tasks.

If standard methods give inadequate performance, there are several avenues that might be promising to explore:

- *Tailoring mathematical reasoning methods to the algebraic structure of the domain:* AMES' knowledge-representation constrains both the kinds of equations that it generates and the kinds of quantities it needs to derive from them. Algebraic techniques that exploit this regularity may be more successful than general methods.

- *Goal-oriented equation generation:* the converse to tailoring algebraic methods to the form of the domain knowledge might be to tailor the application of domain knowledge to specific algebraic goals. For example, a system might begin with a list of attributes for which it must solve, and adapt the way that it applies domain knowledge to produce quantitative models specific to those goals. This might produce simpler equations, and would constrain the number of equations to consider at any one time.

- *Intelligent selection of attribute measurements:* AMES generates its models using fixed choices of coordinate systems and reference frames. There may be cases where more flexibility might simplify the equations that result.

- *Qualitative mathematical reasoning:* the physical reasoning community has produced a number of different methods for reasoning about mathematical systems at higher levels of abstraction. These might be very useful for providing simpler answers to certain queries, and guiding more detailed solution methods for others.

- *Multiple alternative formulations of domain knowledge:* just as having multiple perspectives for analyzing a scenario helps people, so may having modeling alternatives simplify computer reasoning. For example, reasoning about energy and momentum may be easier than reasoning about forces and motions in some cases, even though the formulations are theoretically equivalent.

4.3 More General Reasoning

4.3.1 Problem Overview

The ideal analysis tool takes arbitrary information about a physical situation and uses this to attempt to solve for whatever goals the user specifies. AMES falls short of this model since it requires a highly constrained description of a system's initial configuration, and produces exhaustive information about its evolution from that point onward.

One way this rigid model of reasoning becomes restrictive, as the previous chapter mentioned, occurs when scenarios involve events that have causes outside the domain of expertise, but produce effects within it. Such problems require adapting to whatever information may be available, instead of depending on a fixed set of inputs. They also require using externally supplied information during the course of a simulation, as opposed to solely during its initialization.

Generating exhaustive information about a scenario's evolution can be a disadvantage since it precludes simple solutions to problems that require only a small amount of computation to solve. Furthermore, there may be instances where enough information is present to solve the user's problem, but a simulation fails because it depends on having complete information about the entire scenario.

4.3.2 Research Agenda

In moving toward a more general paradigm for physical analysis, can simulation still play a critical role, since as previously explained, understanding how systems evolve supports many typical analysis tasks. The key, therefore, is to understand when simulations are appropriate, and to provide them with the information they need. As well, it is important to be able to determine if only subparts of a scenario need be simulated: the general area of scenario modeling and problem re-representation is the subject of the next section.

An interesting line of research might be to develop extensions to AMES that can reason about instantaneous configurations of physical systems. The purpose of such extensions would be to allow AMES to take arbitrary information about the configuration of a system at a point in time and derive the information necessary to build a model from this information. This would free AMES from its restriction to highly constrained inputs. One of the challenges of such a project would be to avoid duplicating information by sharing a common knowledge-base with the simulation routines.

4.4 Modeling and Re-Representation

4.4.1 Problem Overview

From the discussion in the previous chapter, it is clear that while the mechanics domain offers a set of very clearly defined canonical models, there are certain behaviors that these models fail to adequately explain. For example, rigid body collisions are outside the scope of traditional mechanics models. In addition, certain things can have multiple alternative models: terrestrial and universal gravitation are both different ways to describe the same phenomenon.

4.4.2 Research Agenda

The fact that models are not perfect suggests that it would be useful for a physical reasoning system to know the limitations of the models it is using, and understand when its answers may be in error. The fact that certain phenomena have multiple models suggests that a program should ideally be capable of selecting models appropriate for particular combinations of scenarios and reasoning tasks.

A program with this capability could also employ problem re-representation as a problem solving paradigm. Already a subject of research [20], this kind of reasoning complements simulation nicely by offering a way to use higher level models (e.g., lever, particle on an inclined plane) to solve common problem types more efficiently.

Another research direction in the general area of modeling might be inventing more accurate representations of physical objects. These could offer a higher level at which to interact with a physical analysis program. One of the program's new tasks would be to find simpler representations of problems for analysis purposes, when such transformations are appropriate.

4.5 Other Areas for Research

The above suggestions for future research, are only a small selection of the issues the current project has raised: they address only the most immediate deficiencies in AMES' design. Other potentially interesting areas to explore include:

- *Intelligent assumptions:* problem descriptions in AMES are much more verbose than their textbook counterparts. This is in part due to the system's inability to use diagrams as input, but even diagrams gain much of their effectiveness from the fact that people make reasonable assumptions about implied or missing information. Automating this behavior is important for simplifying user interfaces to physical reasoning programs.

- *Modeling human physical intuitions:* having a model of the kind of reasoning that people can perform intuitively might be very useful in helping a program design the format of its output. This knowledge would allow a computer to avoid stating the obvious, and help computers to understand what interests people most. Such knowledge would also be crucial for educational applications of this technology.

- *Integrating knowledge about multiple domains:* understanding what domain to consider when reasoning about a high level problem can be a non-trivial task. For example, diagnosing an automobile might use an combination of mechanics, fluid dynamics, chemistry, and both digital and analog electronics. A future challenge will be to develop programs that can effectively use knowledge about multiple domains to reason about such systems.

Chapter 5

Related Research

5.1 Overview

This thesis takes a somewhat unique approach to physical reasoning. Some of its distinguishing features include a focus on domain knowledge representation, special emphasis on algebraic reasoning, a unique perspective on the role of qualitative reasoning, and special attention to a domain that does not lend itself to fixed topology networks of lumped parameter elements. To describe the context of these contributions, this chapter compares the current project with related work in the field.

5.2 de Kleer: NEWTON

Johan de Kleer's Masters thesis, "Multiple Representations of Knowledge in a Mechanics Problem Solver" [10] involved reasoning about particles that slide along one-dimensional "rollercoaster" tracks. The major contribution of this work was a model of how qualitative and quantitative knowledge can be combined to solve physical reasoning problems. Though that work is now somewhat dated, much contemporary research in the physical reasoning field still follows the general paradigm de Kleer explored in that project.

NEWTON, de Kleer's rollercoaster reasoning program, viewed systems at two levels of abstraction: a qualitative level and a quantitative level. The qualitative level of abstraction characterized a particle's behavior in terms such as "sliding along a track", "flying off", and "free falling". Track shapes were qualitatively characterized in terms of the signs of their slopes and curvatures. Domain knowledge about systems at this level of abstraction consisted of information about what transitions between qualitative states were possible.

The quantitative level of abstraction addressed the exact motions of particles over time. Domain knowledge at this level consisted of equations to describe motion in every possible qualitative state, as well as quantitative criteria for disambiguating the state transitions that would occur from a set of possibilities suggested at the qualitative level.

Reasoning in NEWTON proceeded in two stages. First, the system used its knowledge of feasible state transitions to envision the evolution of the scenario and create a tree of possible qualitative behaviors. Then, if that level of detail was insufficient to solve the problem posed to the system, it would use the quantitative knowledge associated with the qualitative states in the envisioning to disambiguate between possible behaviors and determine detailed information about particle motions.

Much subsequent research in physical reasoning has expanded on the qualitative part of this reasoning paradigm, on the observation that in many ways, this kind of reasoning is similar to human physical intuition: a talent that would be useful to duplicate. To this end, there have been numerous attempts to invent qualitative descriptions for various reasoning domains, exhaustively list all feasible transitions between these qualitative states, and reason by envisioning all possible ways particular systems might evolve.

In focusing solely on qualitative aspects of reasoning, however, many systems encounter problems of intractable branching in predicted behaviors. The cause is the lack of detailed information to disambiguate between possible behavior alternatives. In this respect, they miss one of the key lessons from de Kleer's work: the power of having multiple levels of abstraction - being able to obtain a general perspective with qualitative reasoning, and refine these predictions with quantitative reasoning. In this respect, AMES and the algebraic simulation paradigm diverge from the purely qualitative reasoning camp and return to a mixture of both quantitative and a special brand of qualitative reasoning.

Quantitative reasoning lets algebraic simulation reason about highly complex scenarios, involving, for example, multiple contacts and non-inertial reference frames. Using a purely qualitative approach would generate too many possible changes of state to be useful. In addition to allowing analysis of complex scenarios, algebraic reasoning offers a rich set of abstraction capabilities that offer many of the same benefits from generality that qualitative approaches promise.

Although AMES and NEWTON have numerous general features in common, AMES has mechanisms that support much more complex problem scenarios than the older work. Before detailing the features that make AMES unique, it is important to understand how the approaches are similar. NEWTON's quantitative information about qualitative states plays a role similar to AMES' mathematical models of states. Also, the legal qualitative state transitions in NEWTON, and the quantitative conditions that disambiguate them, are similar to AMES' model validity conditions.

What makes AMES unique from NEWTON, however, is that in NEWTON's domain of particles on surfaces, it is possible to enumerate all possible states and transitions, and associate with each state a complete mathematical model. AMES, however, reasons about scenarios that contain arbitrary numbers of different objects in arbitrary configurations. It must therefore assemble models from basic mechanics principles. This necessitates a more sophisticated representation of physical knowledge, and motivates AMES' dependence on quantitative analysis for reasoning about state change.

These demands are responsible for some of the other advantages to AMES' approach to physical reasoning. Perhaps the most significant is that AMES model-oriented scheme for domain knowledge representation proves quite transparent: the reasons for representing each kind of qualitative behavior and the possible transitions between behaviors rest on the firm theoretical foundations of their roles in the modeling process, and have explicit representation in the model component representations. Such regularities may also simplify the knowledge engineering process, giving guidance for identifying and representing behaviors in physical domains.

5.3 Kuipers: QSIM

Another highly influential piece of research in the physical reasoning domain has been the Qualitative Simulation technique by Benjamin Kuipers [21]. Also in a similar spirit to this work, are other paradigms such as Kenneth Forbus' Qualitative Process theory [14]. These and methods like it deal with qualitative reasoning at the mathematical level: they generate high level descriptions of how systems, described by abstractions of differential equations, evolve over time. While these methods have been applied to reasoning about mechanics systems, they are very different in scope from AMES.

AMES' primary focus is constructing mathematical models from physical descriptions, and determining the limitations on their validity. In this research, I largely ignored quantitative reasoning issues, assuming that standard algebraic techniques could be adapted to AMES' needs. QSIM, on the other hand deals only with systems already described by mathematical models.

Another difference between the methods lies in the type of information they consider "qualitative". Qualitative states in QSIM correspond to intervals over which variables are between particular pairs of distinguished "landmark" values. Qualitative states in AMES are periods over which a single set of equations describes a physical system.

These differences are not a criticism of QSIM, however. They are aimed solely at clarifying the differences between it and AMES. This is important especially in light of the myriad of different ways that various paradigms in physical reasoning are "qualitative" in nature. AMES and QSIM actually have a very complementary relationship. AMES focuses on building mathematical models, while QSIM's specialty is high level interpretation of their behavior.

5.4 Forbus: CLOCK and FROB

Kenneth Forbus' work on spatial reasoning shares certain perspectives with AMES, but differs greatly in many other respects. In research on his CLOCK [12] and FROB [15] projects, Forbus advances what he terms a Metric Diagram/Place Vocabulary (MD/PV) approach to spatial reasoning. This perspective is based on the conjecture that there is no general purpose qualitative representation for spatial information. This necessitates two things: task-specific qualitative representations, and quantitative reasoning to support them.

The inadequacies of qualitative representations also motivated AMES's heavy use of quantitative methods. The MD/PV approach and AMES differ radically in most other respects, however. While both methods use a mix of qualitative and quantitative information, they employ it in very different ways. In the MD/PV approach, qualitative states are abstractions defined in terms of quantitative criteria, but the approach gives no guidelines as to what quantitative features are useful to abstract. In contrast, mathematical modeling considerations guide the design of AMES' qualitative states.

In terms of how the two perspectives view reasoning about physical systems, examples of the MD/PV approach in Forbus' FROB and CLOCK projects exhibit patterns of reasoning along the lines discussed in the section on NEWTON. FROB reasoned about the highly restricted domain of bouncing balls in an environment containing simple fixed obstacles. Although it employed a mixture of qualitative and quantitative reasoning, interactions in the domain were simple enough for methods similar to those in NEWTON to work.

The CLOCK project exhibited only qualitative reasoning. As mentioned in the section that discussed NEWTON, AMES differs from this paradigm since it derives its state transition information from quantitative reasoning about mathematical models, and the ranges over which they hold. Like AMES, however, the CLOCK project reasoned about changes in contact configurations. An interesting feature of AMES's approach, however, is that the model components associated with contact give explicit justification for why it is useful to look at mechanics in these terms.

5.5 Novak: Physics Problem Solving Project

Gordon Novak and his students at the University of Texas at Austin have been working on various aspects of reasoning about mechanics, including a reasoning paradigm that depends on problem decomposition and re-representation as its primary feature [19]. Domain knowledge in this scheme consists of stored solutions to primitive problem types, legal re-representations, and heuristics for selecting useful decompositions and problem re-representations.

The advantages this approach offers is that it allows one to give a system reasoning abilities at different levels of abstraction by giving it solved problems at multiple levels. This approach could help reduce the amount of computation required to understand commonly encountered complex problems. In contrast, AMES always builds models from its library of primitive model components. The reasoning task therefore grows with the complexity of the physical scenarios under analysis.

The strength of AMES' focus on always applying basic mechanics principles, however, is that it achieves wide coverage of the domain. It can always generate models to describe scenarios composed of objects that it understands. On the other hand, it is much less clear what portion of a domain that a particular set of stock problems solutions covers, and whether it is possible for a practically sized set of stock solutions to cover a useful range of behavior.

Another problem with the problem decomposition and re-representation approach is that it is currently unclear what methods can identify appropriate problem transformations. Without these, the paradigm is not very effective. Nevertheless, it remains an interesting approach for research: one that is quite complementary to that used by AMES. Powerful problem solving behavior might come from a system that could effectively decompose and re-represent physical reasoning problems, solve any subproblems that match stock solutions, and apply algebraic simulation to those that remain.

5.6 Addanki, Falkenhainer, Nayak: Modeling

Since much of the research surrounding AMES focused on constructing mathematical models of physical systems, it is important to note how the project is different from other work that shares this same general task description [1] [11] [2] [24].

There are two major ways that AMES differs from most research in modeling. The first major difference is that much of the research on building models focuses on techniques for selecting appropriate models for reasoning about particular phenomena. For example, digital and analog circuit designers often use different transistor models. The reason is that the two groups exploits different aspects of transistor behavior. Choosing an appropriate model for analysis is therefore an important step in physical reasoning.

Appropriate models must describe all aspects of the phenomena that one is interested in, with the amount of accuracy that problems require. The most comprehensive and detailed models, however, may be computationally expensive to use, and may obscure high level features.

AMES does not address the model selection task since each kind of object and interaction in its domain has only a single canonical model of behavior. AMES and other works on modeling do, however, share similar approaches to assembling mathematical models for physical systems: they all work by composing models of the individual behaviors and interactions of systems' constituent elements. A consequence of this common approach is that AMES' model component system for domain knowledge representation is similar to schemes used in related research [11] [24].

What distinguishes AMES' modeling task from similar research, however, is its focus on spatial reasoning. In many domains, for instance elementary electrical circuit analysis, the task of building models of physical systems is quite simple: each component has its own model, and the model of the system as a whole consists of all the component models, plus uniform equations that describe the connectivity of the components. Basic models for many mechanical devices also display this kind of simplicity, due to the regularity of their engineered behavior.

The spatial reasoning in introductory mechanics scenarios produces interesting complexity in two ways, however. First, in this domain, each object has a large number of different ways to interact with other objects. For example, determining the effects of a contact between rigid bodies depends on the exact geometry of that contact. In contrast, electrical components typically interact through limited numbers of ports. Second, the patterns of interactions in classical mechanics are in general constantly changing. As objects move, interactions such as contact appear, disappear, and change character. This necessitates reasoning that can parameterize models to cover ranges of different behavior, understand the limits of those models, and update them when change is necessary. Much of what makes AMES and the algebraic simulation paradigm interesting relates to the way they address these demands.

5.7 Sacks and Joskowicz: Computational Kinematics

Research on "Computational Kinematics" [25] by Sacks and Joskowicz shares many features in common with AMES; however, their focus was much more specialized. The goal of that project was to create kinematic models of mechanical devices. This is different from the scope of AMES' task since AMES deals with unconstrained configurations of objects, and uses complete reasoning about dynamics.

The methods that Sacks and Joskowicz developed can generate mathematical models that describe the motions of the parts of a wide variety of mechanisms from their geometry. Another similarity to AMES, in approach, is that they create models of systems by composing models of their components' behaviors (for mechanism kinematics, these behaviors are pairwise motion constraints). Furthermore, like AMES, their methods handle the changes in models that result from change in mechanical part contact configurations.

Since the domain of mechanism kinematics, though complex, is a highly constrained subset of mechanics, the methods they used are difficult to compare to AMES except at the highest level. Reasoning efficiently about mechanism kinematics requires special exploitation of the constraints that engineers design into their machines. Therefore, while the approach used in AMES is quite general, it is impractical for reasoning about the types of the problems that Sacks and Joskowicz attacked in their project. Conversely, methods for reasoning about machine kinematics are too specialized to be of much use in reasoning about unconstrained classical mechanics problems.

Chapter 6

Conclusions

In my opinion, this thesis contributes to the field of physical reasoning in two ways. First, it presents a powerful paradigm for predicting physical behaviors that addresses some of the most important limitations in the qualitative reasoning methods that have become popular in recent years. Second, this project offers several insights into formally representing knowledge about mechanics.

The algebraic simulation paradigm demonstrated in AMES is significant in several ways. One of its most important contributions is that it offers a fresh perspective on the roles of quantitative and qualitative reasoning. Qualitative reasoning in algebraic simulation is a method for constructing mathematical models, while quantitative reasoning handles inferences about systems' evolution. This architecture allows programs to reason about highly complex interactions, without the crippling ambiguity that purely qualitative approaches typically encounter. At the same, time, algebraic simulation retains many of the key benefits of qualitative reasoning. For example, algebraic methods allows algebraic simulations to abstract over ambiguities in scenario descriptions, and generalize over ranges of parameter values.

Another significant contribution of the algebraic simulation paradigm is that it presents a method for constructing mathematical descriptions of physical systems in domains, like mechanics, that cannot be described by fixed topology networks of lumped parameter elements. The features of algebraic simulation that support this power include: a modular decomposition of physical knowledge that reflects the structure of mathematical models in the reasoning domain; and the ability to both predict the limits of scenario models and update them when their evolution crosses these boundaries.

In terms of the significance of this thesis in providing insights into effectively capturing mechanics domain knowledge, perhaps the most important contribution is the model component representational framework. Lucid representations result from the demands of both describing physical behaviors in a modular fashion, and using these descriptions to construct mathematical models of physical systems. Model components make explicit the situations in which represented behaviors arise, the system attributes that such behaviors influence, and the precise relationship that exists between those attributes.

In addition, the discussion surrounding the AMES program offers several specific ideas for capturing and organizing mechanics knowledge. The program's knowledge-base, in particular, gives special insight into characterizing rigid body dynamics: an important subset of the domain.

Looking toward the future, work on this project has raised a large number of issues in physical reasoning, especially the automated analysis of mechanics. In the immediate future, an interesting project would be to construct a program that incorporates many of the suggestions for improvement to AMES. Such a project would be able to evaluate and expand upon the suggestions this thesis makes for representing additional mechanics knowledge; examine efficiency issues in reasoning about the mathematical models that simulations generate; and explore how to apply simulation to various classes of problem solving. Other areas for future exploration could include model selection, problem decomposition, and applications of simulation in design.

To conclude, therefore, this thesis presents an initial look at a paradigm for physical reasoning that combines qualitative and quantitative reasoning in a somewhat novel fashion that offers interesting advantages over methods that are currently popular. This technique is especially useful for reasoning about the complex interactions present in domains such as classical mechanics, and research on this project has generated several suggestions for how knowledge about that domain can be represented in an effective manner.

Bibliography

[1] Sanjaya Addanki, Roberto Cremonini, and J. Scott Penberthy. Graphs of models. *Artificial Intelligence*, 51(1-3):145–177, October 1991.

[2] Jonathan Amsterdam. Automated qualitative modeling of dynamic physical systems. Technical Report MIT/AI/TR 1412, Massachusetts Institute of Technology, January 1993. Revised version of Ph.D. thesis.

[3] Paul R. Cohen Avon Barr and Edward A. Feigenbaum. *The Handbook of Artificial Intelligence*, volume 4. Addison Wesley, Reading, MA, 1989.

[4] Ferdinand Pierre Beer and Elwood Russell Johnston. *Vector Mechanics for Engineers : Statics and Dynamics, 4th ed.* McGraw-Hill, New York, 1984.

[5] Daniel G. Bobrow, editor. *Qualitative Reasoning About Physical Systems.* MIT Press, Cambridge, MA, 1984.

[6] Eugene Charniak. CARPS, a program which solves calculus word problems. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 1968.

[7] Michelene M. Chi, Paul J. Feltovich, and Robert Glaser. Categorization and representation of physics. *Cognitive Science*, 5(2):121–152, 1981.

[8] Shang-Ching Chou and Xiao-Shan Gao. Ritt-wu's decomposition algorithm and geometry theorem proving. Technical Report TEXAS/AUSTIN TR-89-09, Univ. of Texas at Austin. Dept. of Computer Sciences, Mar 1989.

[9] Ernest Davis. Order of magnitude reasoning in qualitative differential equations. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning About Physical Systems*. Morgan Kaufmann, San Mateo, CA, 1990.

[10] Johan deKleer. Qualitative and quantitative knowledge in classical mechanics. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 1975.

[11] Brian Falkenhainer and Kenneth D. Forbus. Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51(1-3):95–143, October 1991.

[12] Kenneth Forbus, Paul Nielsen, and Boi Faltings. Qualitative spatial reasoning: The clock project. *Artificial Intelligence*, 51(1-3):417–471, October 1991.

[13] Kenneth D. Forbus. Qualitative kinematics: A framework. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning About Physical Systems*. Morgan Kaufmann, San Mateo, CA, 1990. Originally appeared in *Proceedings of IJCAI-87*, 430-436.

[14] Kenneth D. Forbus. Qualitative process theory. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning About Physical Systems*. Morgan Kaufmann, San Mateo, CA, 1990. Originally appeared in *Artificial Intelligence* 24: 85-168, 1984.

[15] Kenneth Dale Forbus. A study of qualitative and geometric knowledge in reasoning about motion. Master's thesis, Massachusetts Institute of Technology, Cambrdige, MA, February 1980.

[16] Andrew Gelsey. Artificial intelligence research issues in computational simulation of physical system behavior. Technical Report RUTGERS CAP-TR-8, Rutgers University. Dept. of Computer Science., Apr 1992.

[17] Choon P. Goh. Model selection for solving kinematic problems. Technical Report MITAI-TR-1257, MIT, Sept 1990.

[18] Daniel Kleppner and Robert J. Kolenkow. *An Introduction to Mechanics*. McGraw-Hill Book co., New York, 1973.

[19] Hyung Joon Kook. A model-based representational framework for expert physics problem solving. Technical Report TEXAS AI89-103, Univ. of Texas at Austin. Artificial Intelligence Lab, May 1989.

[20] Hyung Joon Kook and Gordon S. Novak. Representation of models for solving real-world physics problems. Technical Report TEXAS AI88-93, Univ. of Texas at Austin. Artificial Intelligence Lab, Dec 1988.

[21] Benjamin Kuipers. Qualitative simulation. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning About Physical Systems.* Morgan Kaufmann, San Mateo, CA, 1990. Originally appeared in *Artificial Intelligence* 29: 289-388, 1986.

[22] Jill H. Larkin, John McDermott, Dorothea P. Simon, and Herbert A. Simon. Models of competence in solving physics problems. *Cognitive Science*, 4(4):317–345, 1980.

[23] Tomas Lozano-Perez. *Spatial Planning: A Configuration-Space Approach.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1980.

[24] P. Pandurang Nayak. *Automated Modeling of Physical Systems.* PhD thesis, Stanford University, September 1992.

[25] Elisha Sacks and Leo Joskowicz. Computational kinematics. *Artificial Intelligence*, 51(1-3):381–416, October 1991.

[26] Bela Imre Sandor. *Engineering mechanics, statics and dynamics.* Prentice-Hall, Englewood Cliffs, N.J., 1983.

[27] Gerald J. Sussman and Richard M. Stallman. Heuristic techniques in computer aided circuit analysis. *IEEE Transactions on Circuits and Systems*, CAS-22(11), 1975.

[28] Daniel S. Weld. *Theories of comparative analysis.* MIT, Cambridge, MA, 1990.