July 1987

Toward Reasoning Methods for Automatic Mechanical Repair

Pearl Pu
University of Pennsylvania

Toward Reasoning Methods for Automatic Mechanical Repair

Abstract

A knowledge representation scheme, QUORUM (Qualitative reasoning Of Repair and Understanding of Mechanisms), has been constructed to apply qualitative techniques to the mechanical domain, which is an area that has been neglected in the qualitative reasoning field. In addition, QUORUM aims at providing foundations for building a repair expert system.

The problem in constructing such a representation is the difficulty of recognizing a feasible ontology with which we can express the behavior of mechanical devices and, more importantly, faulty behaviors of a device and their causes. Unlike most other approaches, our ontology employs the notion of force and energy transfer and motion propagation. We discuss how the overall behavior of a device can be derived from knowledge of the structure and the topology of the device, and how faulty behaviors can be predicted based on information about the perturbation of some of the original conditions of the device. Necessary predicates and functions are constructed to express the physical properties of a wide variety of basic and complex mechanisms, and the connection relationships among the parts of mechanisms. Several examples analyzed with QUORUM include a pair of gears, a spring-driven ratchet mechanism, and a pendulum clock. An algorithm for the propagation of force, motion, and causality is proposed and examined.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-87-64.

TOWARD REASONING METHODS FOR AUTOMATIC MECHANICAL REPAIR

Pearl Pu

MS-CIS-87-64
GRAPHICS LAB 16

**Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104-6389**

July 1987

Acknowledgements: This research is partially supported by NASA Contract NAS9-17239, Lockheed Engineering and Management Services, NSF CER Grant MCS-82-19196, NSF Grants IST-86-12984 and DMC-85-16114, and ARO Grants DAAG29-84-K-0061, DAA29-84-9-0027 including participation by the U.S. Army Human Engineering Laboratory.

UNIVERSITY OF PENNSYLVANIA
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
SCHOOL OF ENGINEERING AND APPLIED SCIENCE

TOWARD REASONING METHODS FOR AUTOMATIC MECHANICAL REPAIR

Pearl Pu

Philadelphia, Pennsylvania
July 1987

Norman Badler
(Advisor)

(Graduate Group Chair)

Abstract

A knowledge representation scheme, QUORUM (QUalitative reasoning Of Repair and Understanding of Mechanisms), has been constructed to apply qualitative techniques to the mechanical domain, which is an area that has been neglected in the qualitative reasoning field. In addition, QUORUM aims at providing foundations for building a repair expert system.

The problem in constructing such a representation is the difficulty of recognizing a feasible ontology with which we can express the behavior of mechanical devices and, more importantly, faulty behaviors of a device and their causes. Unlike most other approaches, our ontology employs the notion of force and energy transfer and motion propagation. We discuss how the overall behavior of a device can be derived from knowledge of the structure and the topology of the device, and how faulty behaviors can be predicted based on information about the perturbation of some of the original conditions of the device. Necessary predicates and functions are constructed to express the physical properties of a wide variety of basic and complex mechanisms, and the connection relationships among the parts of mechanisms. Several examples analyzed with QUORUM include a pair of gears, a spring-driven ratchet mechanism, and a pendulum clock. An algorithm for the propagation of force, motion, and causality is proposed and examined.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Problem

How do people reason about the way mechanical devices work? How can computer systems be built to simulate such reasoning? How can such reasoning be further applied to mechanical repair automation?

These are some open questions at the intersection of a number of fields: computer applications in engineering, artificial intelligence, and cognitive psychology. Recent work in computer applications in engineering not only has advanced in its method, but also has extended its dimension. One new method is to incorporate artificial intelligence techniques in constructing more *intelligent* CAD (computer-aided-design) systems. In fact, the trend in this area is to provide computer-*automated* systems rather than just design aid tools. In addition, researchers are also trying to cover a broader spectrum of engineering activities. In the area of very-large-scale-integrated circuits, for instance, computer programs are being developed that handle automatic test generation for newly designed chips, automatic diagnosis for detecting chip failures, and automatic chip layout.

Artificial intelligence and its major subfield, expert systems, aim at developing more encompassing techniques and at covering broader areas of human activity. Expert systems are built to simulate human expertise which can handle medical diagnosis, electronic circuit troubleshooting, or nuclear power plant maintenance, to cite a few examples. Research in this area has resulted in new approaches in constructing expert systems. The first generation, characterized by its use of rules as its internal representation, was found to be lacking in "common sense." The knowledge base of early expert systems tended to be either too large or incomplete in complex domains. Therefore, it is preferable for an expert system to use a *model* of the domain and reason by means of this model to generate information. Such a method of reasoning has been called *causal* or *deep* model reasoning, which marks the second generation of expert systems.

Researchers in cognitive psychology are concerned with the investigation of various human problem-solving activities, the characterization of these activities, and the construction of mental models for them. Consequently, the mental models serve as paradigms for the development of expert systems.

Developments in CAD, expert systems, and cognitive psychology have resulted in a new research area, variously called *qualitative reasoning*, *qualitative physics*, *common sense reasoning*, or *deep model reasoning*. Many physical systems in various domains have been studied. Although there is still very little agreement on the definition of qualitativeness and the underlying representation schemes, a central theme is shared by many researchers in this area: humans seem to rely on "common sense" and "qualitative knowledge" more than on quantitative knowledge during problem-solving activities such as the diagnoses, and explanation of physical systems; qualitative knowledge has yet to be further understood.

In the past few years, research projects in qualitative physics have aimed at a better understanding of how a physical device works given the device's constituent parts and their interconnections, and extending the understanding to cover numerous problem-solving areas, e.g., simulation, explanation, and plan evaluation. We noticed, however, that the existing representations can hardly be extended to cover the area of mechanical devices. Mechanical repair is an even more open area for study. For instance, there has been no effort yet at facilitating the reasoning of the consequences when something has to be taken out for repair; that is, the ability to answer the question "what will happen if we take out a part from a device?" Neither is there a representation that supports the reasoning about how to put back a part properly after repair. In a mechanical pendulum clock, for example, the escape wheel is described by [17] [1] as follows:

> The weight which drives the watch is applied to the circumference of the spindle, causing it to rotate. This rotation is, however, arrested by the anchor, which is linked to the pendulum and which periodically engages with, and releases, a toothed wheel called the escape wheel (the combination of escape wheel and anchor is called the escapement). Each time the pendulum reaches its maximum amplitude, one of the projections (called pallets) of the anchor releases a tooth of the escape wheel, allowing this wheel to rotate a corresponding amount. ...

How do we solve the problem of describing the intermittent motion between the anchor and the wheel? What representation scheme will allow us to predict the behavior of the wheel when the anchor is to be removed or if it becomes somehow disconnected from the wheel? More importantly, what should a representation provide in order to capture the subtle geometric information that underlies the connection between a pair of components?

1.2 The Goal

Our research stems from the need to be able to reason about how mechanical devices work and how they behave, and from the need to apply such reasoning to repair.

This thesis investigates how people understand mechanical devices, what knowledge they use, and how commonsense geometrical knowledge has helped them. For the computer to do likewise, we must construct a computational model that supports the reasoning tasks involved in repair. The particular goals of this work are:

1. To define the problem of automatic repair, that is, what the subproblems are;
2. To investigate what type of knowledge is important for repair reasoning;
3. To investigate whether a new representation for capturing such knowledge is necessary.

[1] This two-volume book is an encyclopedia on how things work.

1.3 Overview of the Chapters

Chapter 2 reviews several strands of related work which serves as the foundation of this thesis. Two emphases are made during reviewing: what has and has *not* been done in qualitative physics, and how traditional sciences have treated mechanical design and repair. The background research contributes to the thesis in a number of important ways. The first part of the review leads to the conclusion that a new knowledge representation scheme for the qualitative reasoning of mechanical devices is necessary; the second part of the review gathers information on the methodologies used by engineers to perform design and repair.

Chapter 3 defines some of the assumptions and terminologies which will be used throughout the entire thesis. It then uses these concepts to define the repair automation problem and its subproblems.

Chapter 4 is the core of the thesis. Given that a new representation is necessary to reason about mechanical systems, this chapter gives the requirements and defines the primitive components of such a representation. The representation scheme, QUORUM (QUalitative reasoning Of Repair and Understanding of Mechanisms), is then described in full detail. Three reasoning tasks are also discussed.

In chapter 5, the detailed simulation algorithm is outlined. Several examples are given to illustrate how the simulation algorithm works. More importantly, we use those examples to demonstrate the validity of QUORUM.

Chapter 6 concludes the thesis. There, we discuss what has been accomplished so far, and we propose the next stage of this research.

Chapter 2

Background Review

The theory to be presented in this thesis has evolved from several strands of previous work in different areas. In this chapter, we examine these related researches as the background of our theory and implementation work in the future chapters.

The review is divided into two parts. In the first part, we discuss what has been done in qualitative reasoning so far. We give the review in chronological order so that a perspective may be gained concerning how the theories in this field have evolved. Particular attention is paid to QP theory, developed by Forbus [11]. This theory is used to model changes in physical systems and the consequences of such changes. We would like to show, however, that this theory, like many others, is inadequate to solve the problems pertaining to the mechanical domain that we are interested in, which constitutes the major part of the motivation for this thesis.

The other major part of the review is the result of our investigation on how design and repair are normally done from the engineer's point of view. There, we will examine the concepts and terminologies engineers use in solving design and repair problems. The result of this research also leads to some conclusions on the kind of knowledge that engineers and repair experts use. The next chapter will discuss in detail the characterization of such knowledge.

2.1 Qualitative Reasoning about Physical Systems

2.1.1 Qualitative versus Quantitative Reasoning

Conventionally, scientists reason about physical systems using two quantitative approaches: analytical and numerical methods. Differential equations are used to describe the structure of a system. The behavior of the system is then determined by solving the differential equations, either analytically or by numerical simulation. The analytical solution of a differential equation is derived by obtaining a closed formula. However, it is not always the case that a closed formula is derivable. When this happens, one often uses the numerical simulation method instead. The numerical solution of a differential equation is done by computing the value for the variable on the left side of the equation for the value of each parameter on the right side at each point in time. Such simulation often results in a table of values which can be used to plot a diagram to show the behavior of the system being modeled.

These two quantitative methods are desirable during the design stage of a physical system, in which case the proposed design must be checked in order to detect previously unsuspected landmark values of the system's parameters. Quantitative approaches provide precise and detailed information on a system — once the system has been modeled and an equation obtained, the value of a parameter at any point in time can be computed. There are two major costs, however, to using such methods: in the case of the analytical solution, the method requires a sophisticated mathematical inference method which often fails to produce a closed-form formula; and in the case of the numerical solution, the method requires an interpretation process to construct a meaningful description from its output. Even when the problem is a simple one in the domain, the solution still requires the same amount of effort. In addition to these two costs, choosing the right mathematical model may itself be difficult. Very often the designer has to go through many trial-and-error cycles to obtain the right modeling method.

Qualitative reasoning offers a different approach to the simulation of the behavior of physical systems. Although there is still little agreement on the definition of qualitativeness and the underlying representation, several common characteristics of qualitative reasoning systems are proposed by researchers in this area, such as [2,3,4,5,6,9,12,11,15]: qualitative reasoning systems capture the *understanding* of how physical devices work; provide commonsense to the reasoning of those devices so that simpler versions of a problem can be solved in a simpler way; specify directly the cause-effect relationship; and can reason about how devices work based on partial knowledge. In the following sections, we give a historical review of the development of qualitative physics. Selected works are representative of what we think the current trend is in this field.

2.1.2 Rieger and Grinberg

Rieger and Grinberg were among the first to propose theories in qualitative reasoning. Their knowledge representation consists of *events, tendencies, states,* and *state changes*, related by several different types of *causal links*. They were the first to recognize the need to represent the changes in a physical device in terms of direct cause-effect relationship. Their scheme is able to produce realistic qualitative simulations of the behavior of mechanisms. However, their reasoning method suffers from what is called "shallow model" reasoning in that there is no strong distinction between the structure and the behavior of a mechanism.

2.1.3 Qualitative Process Theory

Qualitative Process theory (QP theory) is claimed to be a general theory on qualitative dynamics. A central idea is that dynamical theories ought to be organized around the notion of physical processes, such as moving, colliding, and flowing. This theory is to be contrasted with classical mechanics in which *dynamics* describes how forces bring about changes in physical systems. For any particular domain, such as particles or fluids, a dynamics consists of identifying the kinds of forces that act between the classes of objects in the domain and the events that result from these forces. Instead of dealing with the different kinds of forces, Forbus has recognized a common object shared by all dynamical theories: process. More importantly, the notion of process also has allowed him to describe changes in a more intuitive, more abstract, and thus more qualitative level.

QP theory reasons about the physical world in qualitative terms in that it has chosen to talk about objects, processes, and causality more directly, much in the same way that humans reason about the world. To the contrary, classical mechanics characterizes the changes in a system by differential equations, which describe how the parameters of objects in the system change over time. The behavior of the system is then determined by solving the equation, either analytically or by numerical simulation. QP theory, however, provides a qualitative language for expressing differential equations.

QP theory aims at a number of reasoning tasks. The major ones are to find out potential processes and to determine activities (i.e., deducing what is happening in a situation at a particular time). For instance, QP theory can predict that if someone heats water in a sealed container, the water will eventually boil, and if he or she continues to do so the container can explode. It cannot, however, tell us the exact temperature, pressure, etc. of the container at a given time.

How does QP theory work? What is the representation being developed? What kinds of inferences does such a representation support? And what are the advantages and disadvantages of the theory? In the following sections, we will discuss the components of the theory in more detail.

Assumptions and Principles

To understand QP theory, we must know what assumptions and major principles are involved. The following paragraphs describe two major concepts in QP theory.

QP theory is developed around the ontology that everything that causes changes in objects is a process. In fact, the central assumption of QP theory is called the *sole mechanism* assumption, namely:

> **Sole mechanism assumption:** *All changes in physical systems are caused directly or indirectly by processes.*

Once again, this assumption is different from those in classical mechanics, which assumes *forces* bring about changes in physical systems.

A principle Forbus sets forth in his theory is called the *relevance principle* of qualitative reasoning, which states as follows:

> **Relevance principle:** Qualitative reasoning about something continuous requires some kind of quantization to form a discrete set of symbols; the distinctions made by the quantization must be *relevant* to the kind of reasoning being performed.

Representation in QP theory

Object Objects are described by *parameters*, which are quantities. When processes affect objects, one can model such effect by changing these parameters. Examples of parameters that can be represented by quantities include the pressure of a gas inside a container, the temperature of some fluid, and the magnitude of the net force on an object.

A quantity consists of two parts, an *amount* and a *derivative*. Amounts and derivatives are *numbers*, which in turn has two parts, *sign* and *magnitude*.

Quantity space The values of a number are not represented explicitly by real numbers, nor a magnitude by non-negative real numbers. In other words, a quantity of an object is not assigned to a particular real number. Instead, quantities chosen to describe some objects in a given situation are arranged in what is called the *quantity space*. A quantity space is a collection of numbers which form a partial order. Elements in the partial ordering form inequality relationships.

The orderings and the elements in a particular quantity space are determined by the comparisons needed to establish certain kinds of facts, such as whether or not processes are acting. It is not clear, however, how a quantity space for a problem in a particular domain can be systematically constructed. Since quantity space will be very crucial in determining activities of a system using QP theory, if the user fails to construct an appropriate quantity space, the modeling of the system will become inappropriate. This is one of the major drawbacks of QP theory.

The orderings and even the elements in a quantity space can change over time in a problem. In fact, changes in orderings reflect the fact that some quantities of objects in a system are changing as the result of some acting processes.

The quantity space provides an illustration of the Relevance Principle; it provides the relevant distinctions because processes typically start and stop when inequalities change.

Individual views The quantity condition and precondition fields in individual views are used to describe the contingent existence of objects. This way, objects can be created and destroyed, and their properties can be allowed to change dramatically. For instance, when we pour water into a cup and then drunk (quantity decreases to zero), the object **contained-liquid** no longer exists. When a spring is stretched so far that it breaks, it is no longer a spring.

Processes Process definition only differs from individual view definition by one additional field: the influence field. Similarly, the quantity condition and precondition are used to determine the contingent existence of a process. We talk about the role of influence next.

Influences According to the sole mechanism assumption, all changes in quantity are caused directly or indirectly by processes. Direct influences are specified in the *influence* fields. It, in a sense, represents the "result" of a process. Indirect influence is represented by using qualitative proportionality relationship, and is usually specified in the *relation* field.

To show the difference between direct and indirect influences, we use the following example. A fluid flow process causes the **amount** of fluid in the source and destination containers change. Therefore, the quantity **amount** is directly influenced by the fluid flow process. We thus use influence field to specify this fact. On the other hand, changes in **volume**, **level**, and **pressure** of the source and destination containers are indirectly caused by the fluid flow process, because these changes are caused by the change in **amount**. We thus use qualitative proportionality to specify this fact.

Another good example that illustrates the correct use of direct influence and qualitative proportionality is to show how QP theory would rewrite the equation $F = m * a$. The

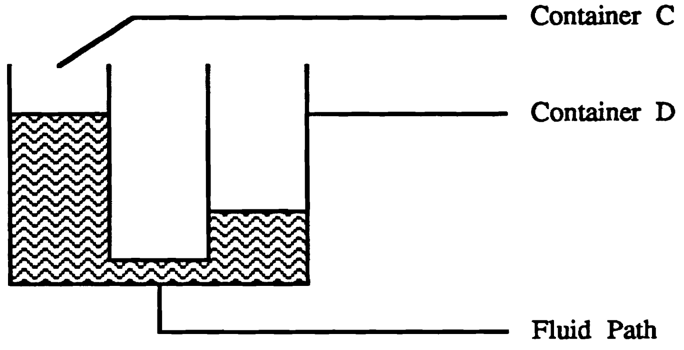

Figure 1: Fluid Flow Process

correct way to rewrite $F = m * a$ is:

$$a \propto_{Q+} F \quad a \propto_{Q-} m.$$

The reason is that we cannot directly apply an acceleration — we can only cause acceleration by imposing a force. It will be wrong to say that mass is qualitatively proportional to force — mass can not be generated by applying more forces; we can only change the quantity of mass by directly adding more mass or deleting them. Similarly, force can only be changed by directly applying more forces in some direction.

What can QP theory do

Forbus claims that QP theory can predict what can happen, describe what is happening in a physical situation, reason about the combined effects of several processes, and predict when processes will start and stop. Let us examine how these tasks are accomplished by QP theory.

In the discussion of the following sections, we use one example to compound the reader's understanding of this theory. The example used is the fluid-flow process, which is shown in Figure 2.1.3.

Finding out what can happen Recall that in specifying a process, one has to specify the individuals the process applies to. When a collection of individuals and a set of process definitions are given, the theory is able to find a collection of individuals that can participate in each kind of process. The theory does so by determining whether or not an individual has met the individual specification in the process definition. If it is met, then there will be a process instance(PI) that relates the individual and the process. In our example, suppose a definition of the **fluid-flow** process is given, suppose two individuals: container C and D are given, suppose both of them contain water, and suppose that there is a fluid path connecting C and D, then QP theory is able to deduce that there are two **fluid-flow** process instances: one process from C to D and one from D to C.

A set of process instances thus defines the *potential* or *possible* processes that can occur among a set of individuals.

Determining activity The precondition and quantity condition fields in a process determine whether or not a process instance is active. The collection of active PIs is called the *process structure* of the situation. The process structure represents "what's happening" to the individuals in a particular situation. For example, if we specify the pressure of the water in C container to be greater than that of the water in D container, then the theory will deduce that the process instance representing fluid flow from C to D is active and that representing fluid flow from D to C is inactive. The answer to "what's happening" is thus there is a fluid flow from C to D.

Reasoning about the combined effects of several processes Most of the changes in an individual are represented by the D_s-values (the sign of the derivative) for its quantities. A D_s-value of -1 indicates the quantity is decreasing, a value of 1 indicates that it is increasing, and a value of 0 indicates that it remains constant. Determining the D_s-value for a quantity is called *resolving* its influences.

Resolving the influences in our example is easy. The fluid flow from C to D is the only cause of direct influences, which causes changes in both **amount-of** for WC and WD. Each of them has only one influence, hence

$$D_s[amount-of(WC)] = -1 \; and \; D_s[amount-of(WD)] = 1.$$

Resolving influences can be difficult, especially when there are a number of processes involved. It is sometime impossible, as Forbus admits. The problem here is common to other qualitative physics theories; it is due to the qualitative nature of the theory, that is, often there is not enough information to perform the necessary inferences.

Finding out when processes stop Changes in quantities can result in changes in processes and view structures. Intuitively this characterizes the disappearance or appearance of processes and individuals. Determining these changes is what Forbus called the *limit analysis*. Limit analysis is carried out by using the current D_s-values and quantity spaces to determine how the quantity spaces can change. For detailed description on how this is carried out, refer to [11].

In our example again, the theory is able to do the following limit analysis for us: the pressures will eventually be equal in the two containers, which means the fluid flow will stop. Hence the process structure set will be empty at the end.

Conclusion

We conclude this section by discussing why QP theory cannot be used to describe the behavior of mechanical devices. More importantly, why this theory cannot be used to describe abnormal behaviors of mechanical devices.

First, as Forbus points out, QP theory is not a language of behavior for physical systems. He argued, however, that it should not be too difficult to extend QP theory to a behavioral language. However, QP theory, like some other qualitative theories (for instance, deKleer's ENVISIONMENT), cannot be used to describe mechanical mechanisms. In QP theory, there is a notion of the relation between quantity and process. However, in order to describe mechanical devices, there has to be a notion of the relation between geometry and motion.

2.1.4 Qualitative Spatial and Geometrical Reasoning

Research in the spatial and geometrical aspects of qualitative physics has been somewhat neglected so far, causing most existing representations to be inapplicable to mechanical devices where such information is crucial to the effective understanding of the behaviors of those devices.

Only recently researchers ([9] [6] [18]) start to investigate the spatial and geometrical aspects of qualitative reasoning. For instance, Gelsey's work takes a solid geometric model of a mechanical device in CSG (Constructive Solid Geometry) forms and from that produces a *kinematic analysis*: a set of mathematical relationships among the positions of the various parts in a device. In doing so, he has reduced the problem of reasoning with geometric relations to the simpler problem of reasoning about algebraic relations.

Some of the concepts defined by Reuleaux [8], such as the *kinematic pair* — a pair of parts which constrain each other's motion — are used in his theory. Kinematic pairs are classified by Reuleaux into two categories: *lower pairs* and *higher pairs*. Lower pairs contact each other continuously at all points on a surface. Higher pairs, on the other hand, contact each other along lines or points. The derivation of the algebraic equations is formed by the following steps:

1. Identify lower pairs by the symmetries of the common surfaces shared by the elements of the pair;

2. Identify higher pairs (particularly gears and cams) by noticing appropriately intersecting motion envelopes;

3. Find constraints on the relative and absolute positions and orientations of kinematic pairs;

4. Detect relationships between the motion of one pair and that of another due to relative geometric configurations of the pairs;

5. Compose these relationships to form new relationships;

The method is highly computational and procedural, since identifying both lower and higher kinematic pairs requires checking the geometry of both objects according to some rules. The fourth and fifth step require finding constraints imposed by a kinematic pair and composing constraints to form new relationships.

Falting's work, in a sense, is more qualitative than Gelsey's. He addresses the problem of *qualitative kinematics*, meaning reasoning about the interactions of objects.

2.2 Related Research in Causal Simulation

In this section, we present a brief discussion of research in this area at the graphics laboratory of the Computer Science department at the University of Pennsylvania. Specifically, we note the work by Paul Fishwick [7] and Steve Platt [14]. Paul Fishwick provides an environment simulating complex systems in a hierarchical fashion and thus a good toolkit for causal reasoning; Steve Platt addresses the problem of representing the knowledge of objects and how such knowledge is to be used in animation. We now discuss each work in more detail.

2.2.1 HIRES

One approach to causal reasoning is to reason about physical systems hierarchically with different levels of detail. Although restricted to simulation only, Paul Fishwick's HIRES provides a good environment for users to *monitor* and *change* abstractions associated with the model while simulating a complex system. He distinguishes abstractions in terms of process, object, and report. The major part of his thesis has concentrated on defining the process abstraction and investigating how interface is accomplished between the different levels of abstraction so that simulation can be done interchangeably among those levels. Consequently, a process, for example, can be represented by a qualitative model, a continuous model, or a discrete model. Three levels of abstraction at different degrees of detail are provided. Simulation can then take place at any of the abstraction levels and switch to any other level.

2.2.2 OASIS

The idea of object/action in OASIS is an attempt to reason about causality. Although Platt's main objective is to be able to model human faces, most of the concepts apply to mechanisms as well. In fact, he starts his research from a mechanical clock, in which he models mechanical parts as objects and the motion of such a clock as actions.

We now examine the object/action paradigm in more detail, since it is very closely related to the way we model mechanisms (a detailed discussion of how we model a mechanism will be presented later in this thesis).

OASIS is an animation system and is broken down into three components: objects, actions, and the animation process. The central concept is the separate definition of those three unities, thus allowing the addition of objects without changing the actions and vice versa. Similarly, the animation process is not designed in terms of any particular objects or actions. Rather, it is designed in terms of accessing fields of the action and applying the action to the object.

To describe an animation system in a high-level environment, Platt further uses generic objects and actions, such as a generic gear, rotation, and translation. Frames are then used to capture the generalized information about an object or action. A set of frames can be created hierarchically, allowing a single frame class to inherit properties from more general frames, and supply default properties to its descendants. In this sense, the actual instantiation of an object inherits the properties of its generic form, which includes how this particular object relates to other objects. Generic actions, on the other hand, know what information they need and how to use that information.

The object/action simulation system thus contains definitions of a number of objects and actions. Those generic objects and actions can be further instantiated to produce a complete representation of the system being animated. The animation process itself merely has to be able to *apply* actions to objects and *resolve* all caused secondary effects. Thus, there is an applicator which is responsible for removing action tuples from what is called the action list, applying the action to the object, and adding any secondary effects back to the list.

Another point worth noticing in OASIS is the way it handles messages. Normally, an object-oriented or frame-based system allows messages to be passed among the objects or frames. In OASIS, messages are not passed between objects; instead, information pertaining to caused actions is sent to the applicator. In effect, this is the hierarchical message-passing approach. A major advantage of this method is to ensure synchronization in the execution of the actions.

2.3 AI and Engineering

2.3.1 Introduction

The science of using computers to assist design, which we call Computer-Aided-Design, has been developing at a very fast pace in recent years.

In the early days, the term "Computer-Aided-Design" was used in a restricted sense to refer to the use of computer graphics in engineering-drafting applications. For instance, through the help of interactive graphics devices, draftsmen were able to sit in front of a terminal creating technical drawings, modifying them, and producing hard copies. Later on some progress was made in achieving highly automated design systems that could really help designers to do design and analysis. The purpose is to eliminate the trial-and-error that one often finds in the old design process. Design analysis, which is time-consuming and error-prone for humans, can now be done by computers more quickly and accurately.

2.3.2 Engineering Design in General

Engineering Design as a deductive activity Engineers and research scientists differ in the cognitive activities that are involved in their work. The principal objective of research that scientists do is the development of models, theories, or hypotheses to describe scientific phenomena. For example, suppose that a scientist observes that the incidence of serious crime tends to increase whenever a long period of hot, dry weather sets in. He might hypothesize that temperature and humidity affect human irritability. The scientist would then set out to verify or disprove this hypothesis by conducting psychological tests of individuals exposed to a controlled temperature-humidity environment.

As one can see, scientific research is an inductive process because one attempts to draw general conclusions from specific experiences.

From: (P a), (P b),...

Infer: (forall (x)(P x))

In sharp contrast, engineering design is a highly specialized process that will perform a required function. Hence it is a deductive procedure that attempts to develop a specific solution to a given problem from a general set of principles. That is, engineer first study the fundamental scientific principles that govern the process of interest and then use theses concepts to synthesize a particular design. Whereas research proceeds from specific experiences to general or abstract principles, design proceeds from general principles and abstract models to specific solutions.

2.3.3 Design Process

The design process can be divided into the following three steps:

1. Feasibility Study;
2. Preliminary Design;
3. Final Design.

The feasibility study requires formulating a variety of general solutions to the design problem and then evaluating the technical and economic feasibility of these solutions.

Once the alternative solutions to a problem have been explored, engineers select the approach that appears most promising and perform a preliminary design. The purpose of the preliminary design is to evaluate the usefulness of the concepts and the ideas incorporated into the design and to determine whether or not the design works.

The preliminary design study usually makes extensive use of models of the design. Such models may take the form of drawings, diagrams, or mathematical calculations. Sometimes these models reveal serious flaws in the design, thus substantial revisions must be made to the design. They may also support the initial design, providing more confidence that the design will actually function as intended.

Once design engineers are satisfied that all foreseeable problems have been eliminated, they then proceed to design the final form of the device or product. The steps involved in the final design are similar to those of the preliminary design. First the final version is specified by drawings. A prototype is then built according to these drawings. In contrast to earlier models, this prototype is identical in all respects to the final product. It performs the same functions. It even has the same size, shape, and color.

2.3.4 CAD's contribution to Engineering Design

As was mentioned, design is a complex process which involves a lot of creativity as well as knowledge of principles. A computer can contribute many improvements to almost every step in the design process. The data base maintained by computers can be used to store past designs, technical literatures and principle knowledge. Powerful modeling systems can be used to model various physical processes and objects, and allow simulation to be performed on those models to achieve accurate analysis. The trial-and-error routine is now replaced by simulation-analysis which is made possible by accurate modeling techniques and computer power.

2.3.5 Where and How AI techniques will make their greatest contribution to engineering design

The first potential benefit is suggested by the significant successes recently recorded in packaging human expertise into 'expert systems' software and putting that expertise into the hands of others. The purpose is to extract engineering expertise and package them in expert systems. Following are some examples:

Some engineering problems require a search through an extremely large space of possible designs. If the search space is extremely large, engineers will be unable in any reasonable amount of time to find even an adequate design. A notable example of such large and complex design spaces is very large-scale integration (VLSI) design. In such cases we can hope to apply the powerful heuristic search techniques of AI to carry out rapid automatic searches through a design space.

A number of the engineering analysis techniques now available as software packages (for example finite-element analysis) are extremely powerful. They are seldom used in practice, however, because most design engineers are not expert enough in the analysis techniques to recognize which problems they are useful for, and do not know how to use them if they wish to. This problem can be alleviated by an intelligent user interface which couples a naive user to a highly sophisticated analysis program.

One of the major issues in CAD systems is the object model representation. Quite a lot of work has been done. Yet new methods are still to come. One of the new techniques from AI is to use logic programming to model object. It allows one to deduce structure from description of the behavior of each of the components in a system. Since design can be viewed as synthesizing structure from behavior, logic programming has potential value in automatizing the design process.

AI techniques further contribute to the development of computer-assisted engineering in the following areas:

- Qualitative simulation and reasoning;
- Representation of causal knowledge;
- Natural language.

2.4 Mechanics and Mechanisms

We have just shown in the previous section how engineers have treated the design issue. To construct reasoning systems in a certain domain, it is important that we incorporate existing knowledge in that domain and, at the same time, characterize and classify the knowledge. In fact, according to Feigenbaum, "the power of a knowledge-based system does not derive from the particular formalisms and inference schemes it employs. Rather, the degree of richness, usefulness, and depth of the knowledge determines the capability of such system." We examine how mechanical engineers have treated mechanisms in this section.

Two classic text books on mechanism, Reuleaux [8] and Schwamb [16], are useful. We will quote a large amount of definitions on some of the most important concepts in mechanisms from those two books. We divide the definitions into three categories: general terminologies and classification of pairs of elements.

2.4.1 General Terminology

The science of mechanism treats the laws governing the *motion* of the parts of a machine and the *forces* transmitted by these parts. The designing of mechanisms is further divided into two parts of study:

1. **Pure Mechanism or Geometry of Machinary**, which treats the motion and forms of the parts of a machine, and the manner of supporting and guiding them, independent of their strength;

2. **Constructive Mechanism**, which involves the calculation of the forces acting on different parts of the machine, the selection of materials as to strength, durability, and other physical properties in order to withstand these forces, taking into account the convenience for repairs and facilities for manufacture.

A Machine is a combination of resistant bodies so arranged that by their means the mechanical forces of nature can be compelled to produce some effect or work accompanied with certain determinate motions.

No machine can move itself, nor can it create motive power; this must be derived from *external sources.* A common example of a machine is an engine.

A Mechanism is a combination of rigid bodies so arranged that the motion of one compels the motion of the others, according to a law depending on the nature of the combination.

There is a small difference between a machine and a mechanism. A combination is a mechanism if it is used to *transmit* or *modify* motion and a machine if energy is transferred or work is done. Thus, a machine is a series or train of mechanisms but a mechanism is not necessarily a machine.

Frame. The frame of a machine is the structure that supports the moving parts and regulates the path, or motion, of them. Often a frame would have a motion of its own.

Driver and Follower. That piece of a mechanism which causes motion is called the driver, and the one whose motion is effected is called the follower.

Modes of Transmission. If the action of natural forces of attraction and repulsion is not considered, one piece cannot move another unless the two are in contact or are connected to each other by some intervening body that is capable of *communicating* the motion of the one to the other.

Thus motion can be transmitted from driver to follower:

1. By direct contact: sliding, rolling;

2. By intermediate connectors: rigid, flexible, fluid.

If an intermediate connector is rigid, it is called a *link*, and it can either push or pull, as the connecting rod of a steam engine. Pivots or other joints are necessary to connect the link to the driver and follower.

If the connector is flexible, it is called a *band*, which is supposed to be inextensible, and capable only of transmitting a pull. A fluid confined in a suitable receptacle may also serve as a connector, as in the hydraulic press. The fluid might be called a pressure organ in distinction from the band, which is a tension organ.

2.4.2 Kinematic Pairs

Pairs of elements. In order to compel a body to move in a definite path, it must be paired with another, the shape of which is determined by the nature of the relative motion of the two bodies.

Closed or lower pair. If one element not only forms the envelope of the other, but also encloses it, the forms of the elements being geometrically identical, the one being solid of full, and the other being hollow or open, we call them a *closed pair*, also a *lower pair.* In such a pair, *surface contact* exists between the two members.

The class of closed pairs are divided into three categories:

1. Screw pairs, which allows helictical motion.
2. Revolute pairs, which allows rational motion.
3. Sliding pairs, which allows translational motion.

Higher pairs. If a pair does not enclose each other, rather, the elements are in a point or line contact, it is called a *higher pair.* Gears, ball and roller bearings are examples of higher pairs.

2.5 Traditional Mechanical Repair

The automation of most existing automated tasks requires investigations on how humans have traditionally treated the tasks first. For instance, in building an expert system to perform automatic medical diagnosis, one would investigate the methods and kinds of knowledge physicians use and then formalize such methods and knowledge in order to automate the medical diagnosing task. In building a vision system to recognize objects, one first investigates how humans perceive the world. To automate the repair process, by the same reason, one should investigate on how humans have traditionally treated this task.

In this section, we will examine issues that are of concern to the mechanical repair community, categorize those issues with some structures, and set some directions for computer automation of the repair process. Most of the following discussion was based on an engineers' handbook on maintenance by Higgins and Morrow [10].

2.5.1 Administrative versus Technical Concerns in Repair

In a industrial company, repair services usually require a group of people to form a special department whose task is to attend to the day-to-day problems of keeping the physical plant — machinery, buildings, services — in good operating condition. This group is usually divided into administration and technical staff. The administration staff consists of maintenance supervisors and maintenance managers. Their objectives are to best allocate man power, enforce the execution of repair plans, and maintain schedule discipline. The technical staff consists of people who are concerned with the planning, scheduling, material handling, and the actual execution of each repair job. They must determine the solution to the problem (i.e., how to accomplish the repair job) and at the same time be concerned with the cost, time, and effectiveness of each repair job.

The steps involved in a repair job involves are outlined in the following sections. The duties of the two groups of the people mentioned above are also described.

- The flow of work often starts in *engineering*. The Engineers' duties include outlining job descriptions, preparing complete and good-quality drawings, and communicating with planners and maintenance supervisors.

- Drawings and job outlines go to the *planners* and *schedulers*. The first purpose of planning is to help the mechanic. Planners prepare a description of the way each job is to be performed — who will do the work, the sequence of steps, the material and equipment requirements, and the manhours required. An effective planner plays an important role in saving the cost, reducing the time, and increasing the effectiveness of the job. Schedulers determine the time when a job will be started and the deployment of personnel to perform the work. They also have to solve conflicts in the use of manpower and enforce decisions on the priorities of jobs.

- Complete descriptive packages for each job are then given to the *maintenance supervisor*. He/she decides on whether the plan and schedule will actually be executed. Cooperation with the planners and schedulers are important to the supervisor. If the supervisor doesn't like the planned method and uses his/her own, much of the planning effort is lost.

- When the supervisor is ready, he/she calls on the *materials handling group*. They have to be trained so that they recognize and identify parts and materials without errors. They also have to deliver the materials and parts directly to the job site.

- Watching over the orderly procession of work orders is the *maintenance manager*. He/she has to maintain schedule discipline, minimize outside interferences with the schedule, and improve all phases of planning, scheduling, and execution of work.
- Finally, the execution of the job is carried out by the *mechanics*. They must be trained in various areas to handle jobs like welding, riveting, screwing, disassembling parts and devices, and using various tooling machines.

2.5.2 Categorizing Repair Activities in Three Perspectives

Repair covers a large spectrum of activities. We categorize them in terms of their *scope, complexity*, and *frequency*.

By the *scope* of a repair job, we mean the involvement of people. The repairing of buildings, roads, and industrial plants requires a special maintenance department in the organization, as we have mentioned earlier. The activities involved in such a department usually consist of the administration of the repair personnel, the planning and scheduling of every repair operation, concerns of cost, efficiency, and time, and the control of actual repair operations. This type of repair activity involves more than a craftsman's talent. It is an engineering task. On the other hand, the repair of, for instance, household appliances, clocks, watches, or toys, is more of craft work and requires only one or a few specialists. So a repair crew can range from one or a few repair craftsmen to a team of well-organized people.

Repair jobs can also be distinguished by their complexity. Some are trivial; some are difficult; and some intractable. Some are routine work, and some are original. A repair job can become difficult for several reasons: too many parts need to be repaired; parts cannot be reached directly; devices are difficult to disassemble in order to repair the inside parts. Complexity is also a relative term for each individual repair person. If all the repair persons were expert, then fewer jobs would be complex.

Another criterion that we use to categorize repair jobs is frequency. Starting with the least frequent repair job, we have 1) rebuilding, 2) preventive repair, and 3) emergency repair. *Rebuilding* occurs probably many years after the device is manufactured or the plant installed. It also occurs on demand. Thorough analysis, careful planning, and scheduling are needed before the decision of rebuilding can be made, and operation begun, since rebuilding almost always involves a considerable amount of cost: the shutdown of the plant, discontinuation of the operation of the machineries, and a lot of man power. *Preventive repair* is, in a sense, a methodology that attempts to save cost, time, and even man power that otherwise would have been incurred in rebuilding. The idea is that if preventive repair is always done on time, rebuilding will not be necessary, or will be less frequent. It is not always the case, though, that more preventive repairs result in less cost, time, and man power. Too many preventive repairs can just be as costly as rebuilding. Preventive repair usually occurs periodically on a yearly basis, or once in several years. Again, the decision on how often such jobs should be done needs a careful study on a number of parameters mentioned above. Finally, *emergency repair* is often the most frequent, least expected, and sometimes most difficult job that a repair person encounters. Whether this kind of job is done successfully or not can be vital to the devices that are worth hundreds and thousands of dollars, and to the lives of people. For instance, the repair of a failed mechanical device in a space shuttle, space station, or aircraft is of that nature. Emergency repair is also the most unmanageble job in an organization. One usually does not know in advance the amount of work involved in such a job, and therefore does not have enough time to plan ahead.

Chapter 3

The Automatic Mechanical Repair Paradigm: A Definition

Automatic repair is a new subject in computer science, although not a new word in existing literatures. Repair is often mentioned in the literatures on automatic diagnosis as the next step after diagnosis. It is, however, assumed to be done by humans. Since this thesis is one of the earliest attempts to tackle this problem, we feel that it is necessary to carefully define the terminologies and concepts involved in the repair process and to define the problem itself formally (i.e., in an abstract notion).

Thus in this chapter, we first state some of the important assumptions and define the terminology that will be used throughout the thesis. Then, we define the repair automation (or the repair reasoning) problem, that is, what steps are necessary for an automatic system to complete a repair job.

3.1 Assumptions

This work is concerned with repair only, although a desirable feature to have is the interleaving of diagnosis and repair. The interweaving of diagnosis and repair is defined as follows: after a device is diagnosed, the faulty part is taken out for inspection to confirm or discomfirm the diagnosis. If the diagosis is confirmed, the repair process begins; if it is denied, the diagnosis process is repeated. We did not include this feature in our paradigm. We see it as rather a feature at a higher level, the level which integrates diagnosis and repair. Therefore, we assume that one of the inputs to our system is the correct identification of faults in a device.

We also assume that only the mechanical domain is addressed, that is, we will be primarily concerned with force, energy, velocity, and the position of an object, and the connections among objects. Should there be any electrical unit, or units that operate according to principles other than those from mechanics, we assume that the repair knowledge comes from different theories.

3.2 Terminologies

In this section, we define the terms that are of concern to our definition of the repair problem. The problem itself is described in the next section.

3.2.1 Components of a Mechanical Device

We use the reductionist approach to describe the overall structure of a mechanical device.

In a mechanical device, there is a natural hierarchy which underlies our definition of structure, behavior and functionality. A device is often made of subassemblies, and subassemblies are made of still smaller subassemblies or parts. We refer to those subassemblies and parts as the *components* or *constituents* of a device. These constituents are also called the constituent *members* or *bodies* of a device. We will be using these terms interchangeably throughout the thesis.

The reductionist approach is thus to describe a device's structure, behavior and functionality in terms of the structure, behavior and functionality of its constituents.

3.2.2 Structure of a Component

The structure of a *component* describes two aspects of that component: *substances* and *geometry*. By *substances* we mean the materials the component is made of or may contain, for instance, gas and liquid. By *geometry* we mean the quantitative attributes, e.g., the size, the weight, etc., of a component.

3.2.3 Topology of a Mechanical Device

Since the structural attribute defined above is a local feature about a component, we use the topological feature to define the overall structure of a device. By the topology of a device, we mean the configuration of the constituent parts in the device.

3.2.4 Connection between two Components

For a body to transfer force and energy to another body in a mechanical device, there must exist a physical contact between the two bodies, either by direct or indirect connections.[1] An indirect connection is achieved, for instance, by means of a link. We define the connection relationship to be the geometrical contact relationship between two physically connected bodies or that established by a third body. Some examples of connection include the tooth-matched connection in a pair of ratchet gears, and the link in a crank-shaft mechanism.

3.2.5 Causality

Causality is defined by the American Heritage dictionary as "the relationship between cause and effect." In the domain of mechanical devices, by causality we mean the relationship between the cause of the *motion* and the effect of the *motion* of an object.

Causality obeys the law of locality, that is, for a causality to exit between two bodies, they have to be physically next to each other or connected by means of a connector.

[1]Since we are concerned only with the mechanical domain, magnetic and electrical field forces are not considered.

3.2.6 Function

The function of a component is the *purpose* the component is designed to serve. It is specified as *what* the response is to a stimulus. For example, the hour hand of a mechanical pendulum clock is designed to indicate the time to an observer; the gear wheel is designed to transfer torques to the hour hand so that it will rotate.

3.2.7 Behavior

The behavior of a component captures the *changes* of the component in response to a stimulus. It is specified as *how* the response is related to the stimulus. Using the clock example again, the behavior of the hour hand can be described by the rotation around a point.

The behavior of a device as a whole is determined from the behavior of its constituents and the interconnection relationship of its constituents. The algorithm that generates this global behavioral description is an important part of causal reasoning in describing how devices work. This algorithm will be presented laten in this thesis.

3.2.8 Structure, Behavior, and Functionality

It is important that we distinguish the three concepts of structure, behavior, and functionality and at the same time relate them. Structure is a description of the *intrinsic* property of a component. Unlike structure, behavior and functionality are concepts that we impose on the device in order to describe to others *how* the device works and *what* the device is used for.

3.2.9 Fault

A fault is any structural *deviation* from the design (or the ideal condition) of a device. Sometimes the design specification is also referred to as the *nominal* data. Design specifications should allow deviations within a certain range (the tolerance range). Deviations within the tolerance range are not considered faults.

3.2.10 Diagnosis

Diagnosis is the process of *isolating and identifying* fault(s) in a device. The diagnosis is often viewed as the process of reasoning from behavior to structure, or more precisely, from abnormal behavior to structural defect: given symptoms of abnormal behavior, one is to determine the structural deviations responsible for the symptoms.

3.2.11 Target

A *component*, whether a part or a subassemblage, diagnosed as having faults, is called the *target*.

3.2.12 Functional Equivalence

The definition of the functionality of an object is still vague, since an object can serve different purposes depending on how it is used. Without specifying what criterions is being used, it is difficult to compare the functionalities of two objects. To solve this problem, we define the functionality of an object with respect to a set of *constraints*. For instance, if a block is used as a door stopper, then what matters is the weight of the block and the roughness of the surface. The size and shape of the block are not important. However, when a part has to fit in an assemblage, the size becomes an important factor if a functionally equivalent part is to be selected.

Thus we define the functional equivalence of two objects to be the functional equivalence of those two objects with respect to a set of constraints. If all of the constraints can be satisfied by both objects, then they are functionally equivalent. Constraints can be of any nature: structural, topological, or material. For instance, two objects are structurally identical, but one tolerates heat better than the other. If the constraint is heat tolerance level, then those two objects are not functionally equivalent. On the other hand, they are structurally equivalent.

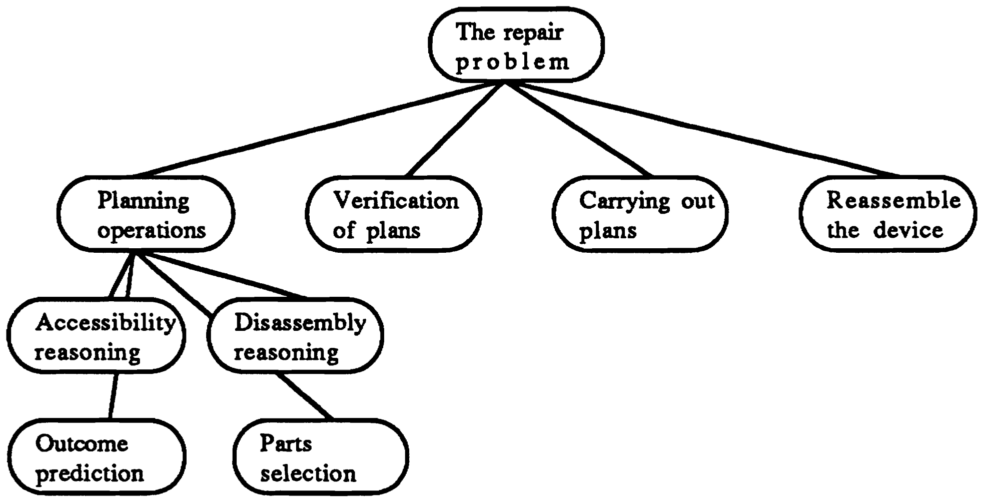

Figure 2: The subproblems in repair

3.2.13 The Goal of Mechanical Repair

Having defined the functionality equivalence for mechanical devices, we thus define the goal of mechanical repair to be the ***achievement*** of functional equivalence of the faulty device and the original (as designed) device with respect to a set of constraints. We emphasize functional equivalence in defining the repair goal here because repair often involves replacement by something which may not be structurally equivalent to its original, but which restores the original functionality.

3.2.14 The Accessibility Problem

In repair, the reasoning about the ***reachability*** of the target object is called the accessibility problem.

3.2.15 The Disassembly Problem

Often the target object is not directly accessible. Disassembly of some connected entities, however, will make the target object accessible. The reasoning about *what* entities to disassemble, and *how* to disassemble them is called the *disassembly* problem.

3.3 The Definition of Automatic Repair

With the terminologies defined, we now investigate the problem of automatic repair, that is, what steps are involved in repair. The ultimate goal in repair is defined above as achieving the restoration of the function of the device with respect to a set of constraints. To achieve this goal,there are a number of subproblems to solve. We enumerate the subproblems as the following:

- Reasoning about the outcomes of the disassembly of an assemblage or subassemblage (e.g., in taking the valve out, water is going to burst out, so one had better shut the water off);
- Determining the operations to perform in order to fix the fault;
- Carrying out the operations determined;
- Selecting parts to replace the faulty ones (this ability should include not only the ability to select the same part to replace the old one, but also the ability to select a different part with equivalent functionality, for instance, selecting a rivet instead of a screw);
- Reasoning about accessibility;
- Knowing how to disassemble a device if the target is not directly accessible;
- Knowing how to reassemble a device after repair;
- Verifying the result of repair according to the repair goal.

The diagram of the repair problem and its subproblems is shown in Figure 2.

These subproblems, if we order them, can be viewed as the steps a particular repair automaton performs to accomplish each repair job. A possible ordering and arrangement of the above enumeration is shown in Figure 3, which can serve as an abstract algorithm for a particular automatic repair system.

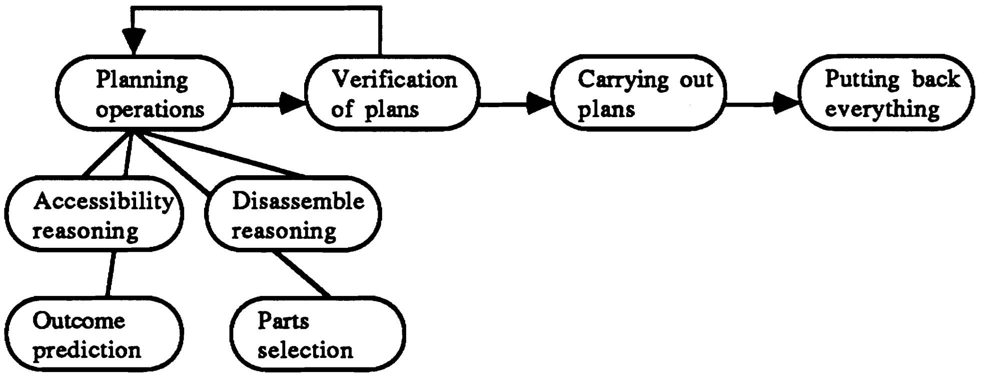

Figure 3: Possible steps in an automatic repair system

3.4 An Example Mechanism

To illustrate of the terms and the paradigm just defined, we now describe an example mechanical device and its structure, behavior, and functionality.

The example chosen here is a pendulum clock, as shown in Figure 4. It consists of five parts: an anchor, a toothed wheel, a spindle, a pendulum bob, and a weight.

The geometrical description is omitted here for brevity. We assume that there are specifications (not shown here) on the size, weight, diameter if applicable, and part number of the five parts, and technical drawings of each of them. Note that those descriptions can be specified in terms of ranges too. For instance, the distance between the two teeth of the anchor has to be a certain length, but the shape does not have to be the shape shown in the picture. The substances that each part of the clock is made of are also omitted. Again, a part can be made of more than one type of material.

The functionality and behavior of each part are summarized in table 1.

To describe how the clock works, we quote a section from [17].

Any periodically repeated phenomenon can be utilized for time measurement,

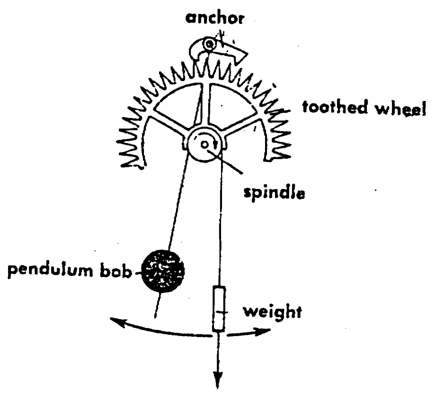

Figure 4: A pendulum clock

name	*functionality*	*behavior*
anchor	controls the speed of the rotation of the wheel	oscillates; releases a tooth for each swing
toothed wheel	drives the minute wheel and hour wheel	rotates around the spindle
spindle	used as a fixed axi	stationary wrt clock
pendulum bob	drives the anchor	oscillates
weight	drives the toothed wheel	attached to a string which is wound on the spindle; moves down as wheel turns

Table 1: The summary of the functionality and behavior of each of the components in the pendulum clock

so long as the duration of the period remains accurately constant. In early timepieces the periodic movement was performed by a pendulum. The weight which drives the watch is applied to the circumference of the spindle, causing it to rotate. This rotation is, however, arrested by the anchor, which is linked to the pendulum and which periodically engages with, and releases, a toothed wheel called the escape wheel (the combination of escape wheel and anchor is called the escapement). Each time the pendulum reaches its maximum amplitude, one of the projections (called pallets) of the anchor releases a tooth of the escape wheel, allowing this wheel to rotate a corresponding amount. Its rotation is therefore performed in a series of jerks, controlled by the anchor and pendulum, and this rotation is transmitted to the hands of the clock through a train of gear wheels. Friction would soon cause the pendulum to stop swinging if it were not given an impulse at regular intervals to keep it in motion, ... In the pendulum clock an impulse is imparted to the pendulum by the escape wheel (which is driven by the weight) through the pallets. The frequency (number of swings per second) of the pendulum can be varied by sliding the bob of the pendulum up or down on its rod. ...

3.5 Possible Questions Raised in Repair

To further illustrate the concerns in repair, we raise some repair-related questions using the pendulum clock example.

1. What would happen if we take the anchor out for repair? (outcome reasoning)

2. What must be removed first, if there is any, in order to access the anchor? (accessibility and disassembility reasoning)

3. If the weight needs to be replaced, what would be an equivalent part? (selecting functionally equivalent parts)

4. What is the geometrical relationship between the anchor and the wheel? (knowledge about the structure of the device)

5. How does the pendulum clock work? (knowledge about how the device functions)

6. How to put back the pendulum rod and bob back properly so that its functionality is resumed? (putting back parts)

They represent some of the questions we are aiming at. Notice while some questions address reasoning tasks, some others address what kinds of knowledge are necessary in repair. Further discussions on knowledge representation issues and reasoning tasks are presented in the next two chapters.

Chapter 4

A Theory on How Mechanisms Work

4.1 Introduction

As we found in Chapter 2, many existing representation schemes from qualitative physics are inadequate for modeling mechanical devices and their behaviors, much less so for reasoning about the repair of such devices. Thus, a new representation is called for. However, what are the requirements of such a representation? What are the primitive components of the representation? What are the reasoning tasks involved?

To answer these questions, we propose a theory on how mechanisms work. Repair experts agree that humans have to acquire knowledge of how mechanisms work before they can repair them. Similarly, an intelligent repair system will benefit greatly if it captures the necessary knowledge of the way mechanisms work. Thus, this theory includes an investigation of how humans acquire knowledge of the way mechanisms work, and a proposal of the requirements for a knowledge representation to capture such knowledge. The new representation scheme, QUORUM (QUalitative reasoning Of Repair and Understanding of Mechanisms), is then proposed and each of its components is discussed. To generate useful knowledge and information concerning how mechanisms work, three algorithms are then

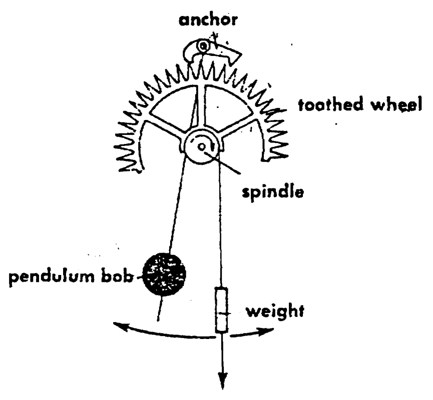

Figure 5: A pendulum clock

proposed.

For the theory to be useful, we impose three requirements on the general characteristics of the representation to be constructed. First, the representation must be able to produce behavioral descriptions of various mechanisms at the qualitative level, that is, in a format similar to that given by a human expert in describing mechanisms to a lay person. Second, the representation must be robust, meaning that it should remain useful in novel situations, for example, when the underlying structure of a device changes slightly. Finally, the representation should support composibility, meaning that a complicated device can be described by its parts and the topology of those parts.

4.2 Describing a Mechanism Using a Causal Model

Our theory is concerned with reasoning about mechanical mechanisms. Pictured in Figure 5 is the same pendulum clock as displayed in Chapter 3, which should illustrate the kind of devices that we are interested in studying.

See page 34 for the description of how the clock works. The description there is from

[17], which is an encyclopedia intended, in the words of the publisher, "to give the layman an understanding of *how things work.*" Let us examine how knowledge from this book will help us in predicting the behavior of the clock.

If the anchor is to be taken away, what might happen? In the description, the author indicated that "this rotation (of the wheel) is, however, *arrested* by the anchor." Thus, if the anchor is removed, the wheel is going to rotate without any control. If the weight is to be taken away, what might happen? Again, since the author told us that "the weight which drives the watch is applied to the circumference of the spindle, *causing* it to rotate," we are able to predict that after the weight is removed, the wheel will stop rotating and the watch will lose its drive.

Notice that the author has implicitly embedded a causal chain in his description; that is, he indicated to us what causes the motion of each of the components in the clock. In fact, most books and repair manuals explain how devices work in this way. Thus if our theory is to *describe* and *explain* how mechanisms work, we propose the following requirement:

> *The theory must generate a description of how devices work in terms of cause and effect.*

How is this description generated? Notice, rules can be used to generate causal-effect relations easily. For instance, we can have something like "if the anchor is removed, then the wheel is going to rotate without control." Indeed, we could easily create a set of formal rules that would produce this ad hoc causal-effect description, but does that description produce a useful understanding?

To answer that question, we examine the robustness of the rule-based approach. In the example above, that rule statement assumes that the anchor is spatially connected to the wheel. In other words, some structural assumptions are embedded in the functionality description of the clock, and will not work for a structurally different clock. The approach violates the "no-function-in-structure" principle set forth by deKleer [3], and thus the approach is not robust. Building a deep understanding requires a representation scheme that is robust, meaning there should be no assumption made about the function of a component at the component level.

For the rest of this section, we discuss the requirements of the components of the representation for describing how mechanisms work in terms of cause and effect. The next section will present the representation in full detail. Following that, we show the reasoning tasks that this representation supports.

4.2.1 Every Mechanism Must Have Some Drive as its External Source

Looking at a mechanism, one often asks "what makes it go?" and "what keeps it going?" In the pendulum clock case, the pendulum bob is likely to be given some initial energy to start the mechanism. Someone either brings the pendulum to a position with potential energy or gives it an initial 'kick'. Furthermore, to keep the the pendulum clock in motion, the author indicated that an impulse is needed. We thus refer what keeps a mechanism running as its *drive*. Sometimes, a mechanism can have several drives.

We recognize those driving sources as the cause of motions. But what is causality? Is there a uniform way of treating them?

Fortunately, if we are dealing with just mechanical systems, there is a uniform way of viewing causality. Newton's first law says that every object preserves its state of rest or uniform motion in a straight line, except in so far as it is compelled to change that state by impressed forces. Thus, force is the only cause for the motion of a single object. An equivalent theory says that if an object is given some energy, whether potential or kinetic energy, the object is also capable of motion.

Causality is spatially continuous between two interacting objects. That is, the two objects are always physically in some kind of contact with each other.[1] Furthermore, causality is directional. In specifying causality, one must indicate what causes what.

Driving sources, which are external to a mechanism, are therefore the ultimate cause(s) of a mechanism. If a mechanism loses its drive, then it will stop running. We thus propose the following requirement of the representation:

[1]We are considering only mechanical systems, so causality obeys the principle of locality.

1. *The representation must explicitly indicate all the external sources for a mechanism.*

4.2.2 Transfer of Force/Energy and Connection Relationship

"The weight which drives the watch is applied to the circumference of the spindle, causing it to rotate." The pendulum oscillates and causes the anchor to oscillate. Mechanisms, viewed in terms of causality, take the form of chain reactions; that is, one object initially starts moving, its motion brings another object into motion, and so on.

For two interacting objects, Newton's third law says that reaction is always equal and opposite to action. When used to analyze how motion is propagated from one object to a secondary object, it is not very useful, however. An equivalent law, the law of energy conservation, is more helpful. It states that when two objects interact and are considered as one system, the change in the potential and the kinetic energy of the system is equal to zero, providing there is no friction consideration.

Therefore, the motion of a mechanism can be viewed as force and energy propagation from one object to another. According to Schwarb [16], "a mechanism is a combination of rigid bodies so arranged that the motion of one compels the motion of the others, ..." In order to assure appropriate propagation of motion throughout the whole system, the *transfer* of force and energy must be achieved in a certain way for every two interacting objects. Thus we have the following requirement of the representation:

2. *The representation must capture for each pair of components* how *the force and energy are transferred, preferably in an input-to-output mapping.*

The connection relationship between a pair of interacting components is the *means* by which force and energy are transferred. For instance, for the pendulum to cause motion in the anchor (or to transfer force and energy properly), a proper connection between them is important. What is the proper connection? This information is, however, omitted in [17] as the author assumes the readers would have some commonsense knowledge. In this clock case, the piece of common sense assumed here is that if two things are to move together, they should be tightly attached by some kind of joint. Thus the following requirement of the representation:

3. *The representation must make all the connection relationships among the components explicit.*

4.2.3 Transformation of Force and Energy and the Nature of a Component

In dealing with behavioral descriptions of a mechanical device, we encounter the problem of deducing what kind of motion a component will exhibit as the result of some input force or energy being applied to that component. Different objects are subject to different motions even though the same amount of force is applied in the same way. A simple example would be to apply the same kind of force to a square block and a cylinder. One is going to translate and the other rotate. The difference between the square block and the cylinder lies in the difference of the *nature* or, more precisely, the physical features of those two objects. Since physical features contribute to how an object is to respond to input force and energy, we view them as transforming input force and energy to a certain type of motion. Thus the following requirement:

4. *The representation must capture how input force and energy are transformed within an object in an input to output mapping.*

In order to fully characterize the behavior of an object, we must also include in our description the physical attributes of the object. Thus the following requirement:

5. *The representation must include all the geometrical features that contribute to the motion of a component.*

4.2.4 Separating the Component Model from the Connection Model

Figure 6 depicts four types of gear trains. All the gears are of the same generic type, that is, they are all made of the same material and shaped in the same way. Assume that

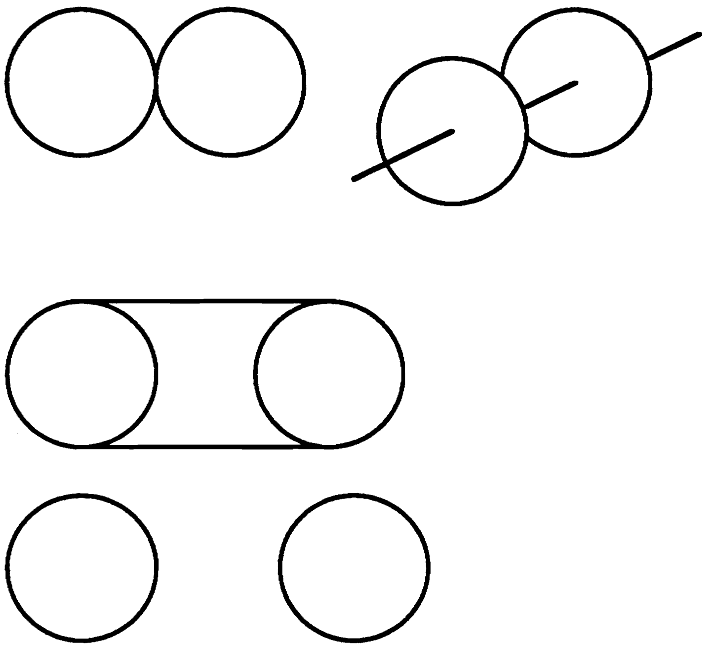

Figure 6: Gear trains

all the gear surfaces have enough friction so that the belt is able to transfer energy from one gear to the other and that when two gears are put next to each other, one gear is also able to transfer energy to the other. The only difference in these gear trains is the way two gears are connected to each other. The way one gear transfers energy to the other is, therefore, totally different for the four pairs, resulting in different motions for the secondary gear.

It is clear now that the connection between two objects carries a lot of functionality, that is, how one object transfers force and energy to another one. If we mix a component with its connection to the surrounding components, we at the same time have embedded some functional descriptions in an otherwise pure structural description. Earlier we had the same problem of violating the "no-function-in-structure" principle by using the rule-based approach. The remedy is therefore to separate structure and topology, that is, to have different representations for describing components and their connections. For the four gear trains, we need only one model for the gear and four connection models for the different types of connections. Thus we propose the following requirement of the representation.

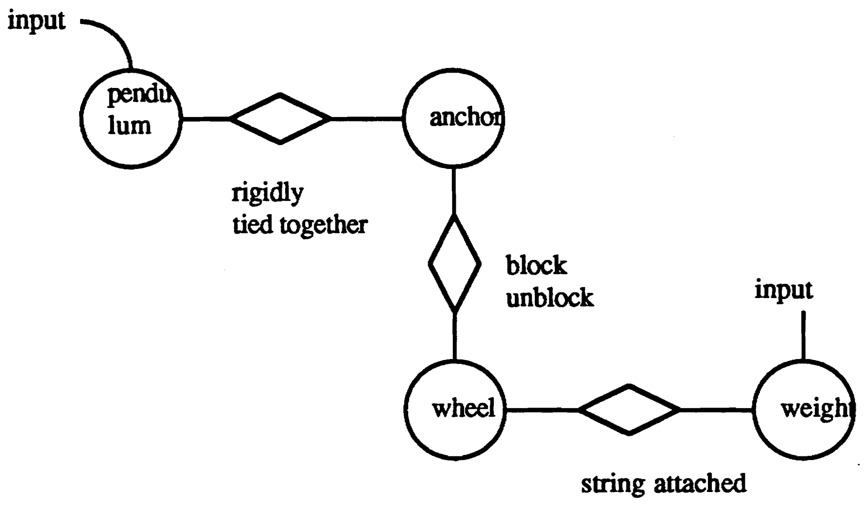

Figure 7: The causal model representation

6. *The representation must make explicit the distinction between the component model and the connection model in order to assure the "no-function-in-structure" principle.*

4.2.5 The Summary Diagram

To summarize, we define what causality means in a mechanism. According to the dictionary, causality is defined as the relation between cause and effect. In mechanisms, there are two kinds of causality of motion: the transfer of motion and the transformation of motion. The geometrical connection between two components defines the transfer of motion. The physical properties of a component define the transformation of motion. Putting everything together, we derive the diagram as show in 7. In the diagram, large circles are components and small circles connections. The arrows represent the direction of causality.

We will discuss the detailed representation in the next section. We do want to point out, however, that this representation is different from Rieger and Grinberg's model [15]. Here the ontology is a clear one: everything in the model is described in terms of either an object or a connection relationship between two objects. In Rieger and Grinberg's model, there are notions about objects, states, and ten different types of links. Furthermore, their purpose was to simulate mechanisms, while we want to do repair reasoning as well.

4.3 The Representation Scheme in Detail

In this section, the representation of QUORUM is presented in detail. By representation, we mean a mapping from the real world to a model with which we can perform knowledge manipulations and answer some questions about the real world.

The major constituents in the representation scheme are the component and connection models, which will be described in full detail later.

4.3.1 The Internal Structure

The basic internal structure of the representation is either a *formula* or a *function.* But what do formulas and functions denote? A formula denotes a "proposition" or "possible state of affairs." Because predicate calculus has the compositional semantics, it is important when introducing a function or predicate to say exactly what the types (and, of course, the number and order) of its arguments are, and exactly what type and denotation of the term it constructs. It may be correct to use the symbol **likes** in a construct like **likes(john, lisa)**, or in a construct like **likes(john, girls)**, but not both. We combine formulas with connectives (the usual **and, or, if, not**) to give new formulas.

Formulas have the following form:

predicate(subject, object)

Some examples are:

rigidly-connected[2](gear1, shaft1)
part-of(bob, pendulum)

[2]Appendix A defines all the predicates and functions used in this thesis.

Functions, on the other hand, denote individuals and actions. They are not in the sense of the function in Lisp; they are not "evaluating" to some individuals. They can be used inside another function or formula. Some examples are:

left-arm(anchor)
move(object, destination)

where **left-arm** denotes a certain part of the object and **move** denotes the action of causing the object to be moved to the destination place by some agent.

4.3.2 The Algorithmic Knowledge

The algorithmic knowledge that is associated with the flow (or transaction) of a mechanism is represented by using the object-oriented paradigm. A mechanism is a network of objects and links. Objects correspond to components and their connections, and links define the topology of the mechanism. In the object-oriented paradigm, messages travel from one object to the other object. Objects in turn, upon receiving a message, will respond by executing some of its methods (or procedures). Similarly, in a mechanism, force and energy flow from a component node (or a connection node) to a connection node (or a component node). Components and connections in turn respond to force and energy. It should not be too hard to see the propagation/action circle that exhibits in both mechanisms and the object-oriented paradigm.

4.3.3 The Component Model

Our representation employs component-oriented ontology.[3] Each part in a mechanism is an object. Each object is represented by a node called the *component node* with a symbolic identification name. To satisfy the "no-function-in-structure" principle, no topological description will be embedded in the component node. Another way of putting "no-function-in-structure" principle is that objects described at the component level should be only generic

[3] A component oriented ontology views everything in terms of an object or a component. An alternative ontology will be process-oriented or state-oriented.

```
Node id :
Individuals :
Properties:
Precondition assumptions:
Physical variables :
Component behavior description :
              Method 1 : input expected:
                         current state:
                         next state :
                         output to:
              Method 2:  input expected:
                         current state:
                         next state :
                         output to:
                         ⋮
```

Figure 8: The component node

objects. For instance, a ratchet gear mounted on a shaft should be described the same no matter what or how other components may be connected to it.

The component node consists of five parts and has the scheme illustrated in figure 8. Now we describe each of the fields in the component node in detail.

Individuals The field *individuals* is a list of component names. The names are merely selected for the convenience of the reader; there is no meaningful semantics attached to them. Often a component node is used to represent a functional unit of many parts rather than a single part. For instance, a pendulum consists of a hinge, a rod and a pendulum bob. Among the many parts in a functional unit, only one is of primary functionality. Others are what is called *supporting* individuals, e.g., a surface or a pivot point, whose function is not to transfer force and energy but to allow the primary part to behave in a certain way.

If, however, the functional unit transmits force or energy through more than one of its parts, the unit needs to be split into two or several component nodes.

Properties Every component has some physical (or intrinsical) properties that serve to distinguish itself from others, such as its shape or weight. More importantly, physical properties of a component contribute to the way the component *responds* to external force or energy, and thus partly determine how the component behaves. Using the same example as before, a square block and a cylinder, both resting on a flat surface, respond to the same external forces differently: one slides along the surface while the other rotates.

In addition, physical properties model the contingent view of a component. A relaxed string has a certain length. It can be stretched to go beyond that length. It can still be stretched so far it breaks. These three strings are different objects, since a stretched string transmits force in one direction while a relaxed and broken one do not. If expressed in predicates, the physical properties correspond to the quantity conditions in Forbus' QP theory[11].

Physical properties are divided into two categories: geometrical properties and material properties. Geometrical properties address features like the shape (e.g., whether a component is a block or a cylinder, etc.), the volume, or the orientation of a component. Material properties pertain to the matters the component of interest is made of. Figure 9 includes the most common geometrical and material features that affect the way an object behaves. Not all of them have to be included in when one is writing the component properties; only those that are applicable to the component of interest need to be considered.

Precondition assumptions Physical (or intrinsical) properties alone cannot determine the motion of an object entirely. The external environment affects the way an object behaves as well. An object behaves differently, for instance, depending whether it is resting on a surface or is suspended in the air by a rivet. To be able to determine the motion of a component, we thus need to know how it is situated in the world (e.g., whether resting on a surface or mounted on a pivot). For instance, a revolving object must have an axle and furthermore the axle must be lubricated so as to allow revolution. The following is an example of the property and preconditional description of a pendulum which is held by a

1. geometrical properties
    - shape
    - length
    - width
    - size
    - volume
    - radius
    - distance between
    - position
    - angle between
    - orientation
2. material attributes
    - weight or friction coefficient
    - elasticity
    - rigidity

Figure 9: Geometrical and Material properties

hinge and is able to swing. [4]

Node id: pendulum

Individuals: bob, rod, hinge, kind-of(hinge, joint)

Properties: weight(bob) > 0

rigid(rod)

Precondition assumptions:

rigidly-connected(bob, rod)

revolution-between(rod, hinge)

held-against-gravity(rod, hinge)

Another example describes a ratchet wheel which is mounted on a shaft and is able to turn.

Node id: wheel

Individuals: wheel, shaft, is-a(shaft, joint)

Properties: shape(wheel, round)

has-teeth(wheel)

Precondition assumptions:

has-pivot(wheel)

held-against-gravity(wheel, shaft)

revolution-between(wheel, shaft)

How a component is situated and shaped is not an accident. It somewhat captures the designer's purpose. Most of this *design knowledge* is, however, lost. Here, we suppose a designer is to sit down and write this field for us. The data collected here is therefore the nominal data, a description of the ideal properties for the component.

Physical variables Knowing the property of a component, how it is situated in the world, and some input stimuli, we can determine the possible behaviors of the component, but not

[4]Appendix A contains a list of predicates and functions, and their semantics.

uniquely. To be more specific about the behavior, we need to know the current state of the component. For instance, in the pendulum example, if the pendulum bob has angular velocity (thus kinetic energy), we can predict that the pendulum will continue to move. But we do not know in what direction the pendulum will move unless some information about the state of the bob is given, for instance, the velocity of bob is clockwise and its current position is the middle position.

Thus, to talk about behavior, we must select some *physical variables* (e.g., position and velocity) that can be used to describe the instantaneous *state* of a component. Such variables can be used to compare the behaviors of two components and also as a place holder to remember the current state of a component in order to allow predictions of the next state to be made.

The state of a component is often defined by a set of position coordinates, plus their derivatives. Thus, state implies *configuration plus velocity*. Configuration tells only where the object is, but state tells both where it is and how fast (and in what direction) it is going. In our representation scheme, physical variables often address the position, velocity, and energy level of a component.

Unlike quantitative variables, physical variables here are qualitative variables and thus can take one of only a small number of values. This set of possible values is determined by the quantity space [11] it participates in. Each qualitative value corresponds to some interval on the real-number line. The most simple, but often used, quantity space consists of only three values: $+$, $-$ and 0. $+$ represents the case when the quantity is positive, 0 represents the case when the quantity is zero, and $-$ represents the case when the quantity is negative. Sometimes we can define $+$ and $-$ according to the actual situation. For instance, we can define clockwise to be positive and counterclockwise to be negative.

Component behavior description In the component node, we are interested only in the local behavior of a component, again without consideration of the interactions of the neighbor components. The behavior of a device as a whole can be derived from the local behavior of its constituents, plus the information on the topology of the device. Such derivation is part of our reasoning tasks and will be shown later among many other algorithms.

Behavior captures how a component responds to an external stimulus. There are many ways, however, to implement the behavior descriptions. In the past, scientists have used the notion of displacement, velocity and acceleration to describe the instantaneous state of a component and furthermore to relate those variables in a differential equation. In a system dynamics course, a student is taught to establish a *mathematical model* for a mechanical device in the following way:

1. Draw a schematic diagram of the system, and define variables;
2. Using physical laws, write equations for each component, combine them according to the system diagram, and obtain a mathematical model;
3. To verify the validity of the model, the performance prediction, obtained by solving the equations of the model, is compared with experimental results. If the experimental results deviate from the prediction to a great extent, the model must be modified. A new model is then derived and a new prediction compared with expected results.

A mathematical model allows us to perform various analyses about the device, for example, through the use of plotting diagrams. Such analysis is often useful in analyzing whether or not a design is valid. The math model is not, however, very powerful for describing the behavior at the intuitive level. Suppose we have an analytical solution of a system and suppose we would plot diagrams which describe the behavior of the system according to the equation. After the data is obtained, we still have to somehow *interpret* the data. For instance, in the case of the angular velocity of a rotating wheel, depending on the magnitude of the velocity, we would have to interpret this quantity and register in our head using qualitative terms such as whether the wheel is rotating slowly or spinning rapidly. Only such terms are useful when we have to assess the situation and decide actions accordingly.

Since our goal is to describe how a device works at the qualitative level, symbolic description of the behavior should be sufficient. We are more interested in descriptive

Component behavioral description:

```
input: torque(gear, DIR) > 0
current state: velocity(gear)=0
next state: velocity(gear, DIR) > 0
output: energy(gear) > 0

input: blocked(gear, SOMETHING, DIR)
current state: velocity(gear, DIR) > 0
next state: velocity(gear, DIR)=0
output: energy(gear) =0

input: energy(gear) < energy_minimum or energy(gear)=0
current state: velocity(gear, DIR) =anything
next state: velocity(gear, DIR)= 0
output: energy(gear)=0
```

Figure 10: The behavioral description of a gear

power than numerical details. As was pointed out earlier, components are the means by which input force and energy are transformed. To capture the transformation directly and intuitively, we employ the input-to-output mapping technique. That is, for a given set of stimuli (or inputs), we map them from one set of states to another set. At the same time, some outputs will be determined in terms of force or energy. Figure 10 is an example of the behavioral description of a gear.

4.3.4 The Connection Model

The connection between two components is the medium by which propagation of motion between the two components is realized. Hence connection defines the precondition for causality; it is the means by which force or energy is transferred. By manipulating the connection, people actually derive different mechanisms from the same building blocks. Using the gear train example in section 4.2.4 again, we can see that because of the different ways a pair of gears is connected, the four sets are used to serve entirely different purposes.

Furthermore, connection is also a problem involving the relation between geometry and motion. Analytical kinematics investigates the quantitative relationship between the connection geometry and the propagation of motion. Gelsey [9] for instance, developed an algebraic method for obtaining equations that relates the motion of one component to the motion of a secondary component. Those two components are called a kinematic pair. A mechanism is viewed as a chain of kinematic pairs in his theory. Our work described the relation between connection geometry and motion propagation at a higher level than was done in his theory. We are interested in a symbolic description of the relation, close to how humans would describe such a relation. Together with the connection characteristics, the causal rule in the connection node is intended to capture the relation of geometry and motion propagation in a more intuitive way.

In addition to the above-mentioned problems, we also try to address the deletion problem here;[5] that is, if the proper connection fails, we cannot just assume that the mechanism will stop working. Undesirable behavior or even dangerous behavior can be the result of a failed connection.

Earlier works [11] [3] in qualitative physics have explored the connection problem, but their notion of connection is too vague to account for the intricate geometry of the connection and the prediction of behaviors of a mechanism after some connections had failed. Forbus, for instance, had something like

$$aligned(source, destination)$$

as a precondition for transfer of heat between a source and destination. Preconditions are only checked initially for satisfaction in order to instantiate an individual view or a process instance (PI) (see chapter 2 for a more detailed review). Once such a fact is established, there is no inference mechanism that will allow us to predict what might happen when source and destination become unaligned in the middle of the course. That is, the condition of being aligned is assumed to be once true and true forever. deKleer uses conduit as the notion of connection. Conduit is described to be "simple constituents which transport material from one component to another and cannot change any aspect of the material

[5]In [4], deKleer and Brown proposed the deletion problem: models should not predict that a machine still functions when a vital part is removed.

within them." However, some of the connections that exist in mechanisms can hardly be modeled as conduits, for example, the connection between the anchor and wheel in the clock example.

Our notion of connection relationships is more elaborated. We distinguish three different types of connections: continuous, intermittent, and impulsive (or one shot). Continuous connections are those that persist until they fail to exist or break down. Most connections in machines are continuous. The connections in the gear trains, for instance, are all continuous. Intermittent connections are characterized by their periodicity; the connections go through a series of phases during a fixed period of time. The anchor and the ratchet wheel connection is a good example. Every time one of the pallets of the anchor reaches its maximum amplitude, the other engages with the wheel and thus blocks the motion of the wheel. When the anchor is in the middle position, its two pallets are not in contact with the wheel. One shot connections are those that happen only once. They often take place at the beginning of the entire course. For instance, when the pendulum bob is given an initial 'kick', a connection is established between the bob and an outside agent. Such a connection, however, only lasts for an instant.

The principle of locality applies to the connection relationship; that is, two interacting components are always physically next to each other. Hence, the connection relationship models not only the interaction between two components, but also the topological relationship. One *connection node* is precisely established per pair of components that interact[6] with each other. It has the scheme illustrated in figure11.

The *node id* is again merely for identification purposes. *Individuals* is a list of names of individuals that participate in the connection. *Connection characteristics* field specifies the geometrical preconditions under which the connection exists. Intermittent connections are hard to specify, since they involve timing. We will discuss how to specify the connection in detail later. For now, let us consider only continuous connections. The last field, *causal rules*, is the most important part of the connection model. It specifies how the motion

[6]Two objects interact if there is some force or energy transfer between them so that one's motion is caused or affected by the other.

```
Connection node id :
Individuals :
Connection characteristics:
Causal rules:
        Method 1 : input expected:
                   output to:
        Method 2:  input expected:
                   output to:
                   ⋮
```

Figure 11: The connection node

of one component affects the motion of the other in an input-to-output mapping. It does so by explicitly indicating how, upon receiving some input force or energy, the connection transfers the input into the motion of the secondary object. The current state is useful in handling intermittent motion transfer, since without knowing the current state, the output cannot be uniquely determined.

Figure 12 illustrates how a connection between a pair of toothed gears might be specified using our representation scheme.

4.3.5 A Mechanical Device is a Network of Component and Connection Objects

Perceiving components and connections between components as objects, we therefore model a mechanism as a network of component and connection objects. The advantage of using the object-oriented paradigm as the underlying control scheme is that it is natural for mechanisms. When a human observes a mechanism, he or she establishes a mantal network as the overall topology of the mechanism. Then taking a closer look at the mechanism, he or she sees individual parts and their interactions. Each of the individual parts is capable of motion once some input force or energy is given, corresponding to the object responding to some input messages. Each of the interactions is transforming force or energy for the next part.

```
Connection node id: toothed-gear-pair
Individuals: gear-1
             is-a(gear-1, gear)
             gear-2
             is-a(gear-2, gear)
Connection characteristics:
            aligned(gear-1, gear-2)
            is-a(connection(gear-1, gear-2), continuous)
            tooth-matched(gear-1, gear-2)
            is-a(connection(gear-1, gear-2), continuous)
Causal rules:
      current state:
      input: energy(gear-1) > 0, velocity(gear-1, DIR) > 0
      output: energy(gear-2) > 0, velocity(gear-2, DIR) > 0
```

Figure 12: The connection between a gear pair

Furthermore, the object-oriented paradigm captures the algorithmic knowledge that one associates with the operation of a mechanism.

4.4 Reasoning with the Representation: Simulation, Prediction and Planning

In the previous section, we have constructed an underlying representation to map mechanisms from the real world to a computational model. The representation will be of very little use unless it supports some interesting reasoning tasks, especially those that are useful in solving repair automation problems. In particular, we are interested in the following reasoning tasks:

1. *Simulation:* Starting with a description of the structure and topology of a mechanical device, deduce the behavior of the device as a whole based on some input in terms of motion, force, or energy.

2. *Prediction:* Given the original structure and topology and a deviation in structure or topology, simulate the behavior. By deviation we mean that some components are missing from the topology, the connection between two components has changed, or the structural property of a component has changed.

3. *Planning:* Given a desirable situation, determine the steps necessary to achieve it. This capability is useful at the post-repair stage when the task is to put everything back together properly, safely, and effectively.

Chapter 5

Examples and Their Simulations

In this chapter, we will select a small group of mechanisms and discuss how to describe them using the representation scheme outlined in the previous chapter. Furthermore, we show a detailed simulation of those mechanisms.

There is a question of whether mechanisms can be classified into a small set of primitive mechanisms and from that set, all the others can be derived. In fact, this is a classical question, well known to the mechanical community. As B. Paul wrote in [13], "Until the time of Reuleaux it was usually accepted that there exists a small number of *simple machines*, which, acting in combination to form so-called *compound machines*, could produce the most general form of mechanical device. However, as pointed out by Reuleaux [1876, p.275], previous writers could not even agree on the number of simple machines, much less their form." Nevertheless, if we classify the major features of existing machines instead of the machines themselves, we might find a manageable set of those features. As Dr. Paul agreed, "But all of us, whether endowed with great or little talent for invention, can profit through familarity with the major features of existing machines."

In the following sections, we first classify the major features of mechanisms in three aspects: the mode of transmission of force or energy between two bodies in a machine, the kinds of motions a body is subject to, and the kinds of joint relationships among the bodies. The purpose is, then, to derive a set of representative mechanisms to use as examples. Finally, we show how to describe the structure and topology of those mechanisms using the representation, and how to simulate their motion.

5.1 Classifying the Features of Mechanisms

5.1.1 Modes of Transmission

If the action of natural forces of attraction and repulsion is not considered, one body cannot move another unless the two are physically in contact, or are connected to each other by some intervening body that is capable of *communicating* the motion of the one to the other.

Thus motion can be transmitted from driver to follower:[1]

1. By direct contact: higher pairs and lower pairs;

2. By intermediate connectors: rigid, flexible, fluid.

Both direct and indirect contact modes can be expressed using the connection node scheme in QUORUM. For indirect modes, an additional individual, the connector(s), should be included in the individual field besides the driver and follower.

5.1.2 Motion Classification

The kinds of motion are listed as follows:[2]

Regular motion is change of position. Motion and rest are necessarily relative terms within the limits of our knowledge.

Intermittent motions. When the motion of a body is interrupted by periods of rest, its motion is called to be *intermittent.*

Continuous motions. When a body continues to move indefinitely in a given path in the same direction, its motion is said to be continuous. In this case the path must return on itself, as a circle or other closed curve.

[1]See 2.4.1 for the definition of drivers and followers

[2]The definition of motions is from [16]

Reciprocating motion. When a body traverses the same path and reverses its motion at the ends of such a path, the motion is said to be reciprocating.

Coplannar motion. A body, or a series of bodies, may be said to have coplannar motion when all their component particles are moving in the same plane or in parallel planes.[3]

Revolution. A body is said to revolve about an axis in the plane that is perpendicular to that axis. The term *rotation* and *turning* are often used synonymously with revolution.

Oscillation is a term applied to reciprocating circular motion, as that of a pendulum.

Translation. A boy is said to have motion of translation when all its component particles have the same velocity, as regards both speed and direction. If the particles all move in straight lines, the body has *rectilinear translation* and, if they move in curved paths, the body has *curvilinear translation.*

5.1.3 Joint Type Classification

Joint types are classified in terms of whether motions are allowed between the joined two bodies or not. According to Paul's classification on joint types of plannar mechanisms in [13], we have the following joint types which allow motions between the joined two bodies:

1. Hinge (a type of lower pairs)
2. Sliding (a type of lower pairs)
3. Gear pairs (a type of higher pairs)
4. Cam pairs (a type of higher pairs)

The other class, which does not allow motions between the joined bodies, is further divided into discrete and integral joint types [1]. When there is a third body involved in the joining of two bodies, such joint types are called discrete joints, for example, rivets and screws. Integral fasteners are formed areas of the component part or parts which function

[3]All the motions considered here are assumed to be coplanner motions.

composing methods	reverse operation
lanced tab joining	unbend, pull, cut
shear-formed tab joining	unfold, cut
embossed protrusions joining	
seam joining	unfold, cut
crimp joining	flatten
screwing	unscrew, cut
bolt/nut joining	unscrew, cut
pin/socket joining	pull apart
nailing	pull out, cut
riveting	chip , cut
quick-release fastening	unfasten
stitching	cut the stitch
stapling	cut the staple
shrink and expansion fit joining	
welding	cut
soldering	unsolder, pull apart
gluing	dissolve the glue, pull apart
covering	lift, pry
enclosing	open
latch joining	open
lock joining	open
hanging	take down
putting on top of each other	take away
bringing together by force	release the force
sliding in	slide out
filling	withdraw

Table 2: The summary of composing methods and their reverse operations

by interfering or interlocking with other areas of the assembly, such as lanced tab joints. Table 2 is a list of discrete and integral joint types.

5.2 Representative Mechanisms

We have chosen a pair of gears, a spring-driven ratchet mechanism, and a pendulum clock as the first group of targets to study the simulation of their motion. This group covers a large subset of the features enumerated above:

- Higher pairs (gears);
- Lower pairs and direct joint (hinge);
- Indirect joint (rod);
- Discrete joint (welding joint);
- Integral joint (screw)
- Intermittent motion (spring-driven ratchet, pendulum clock),
- Oscillation (pendulum bob)
- Revolution (gear)

5.3 The Simulation Algorithm

As we defined in Section 4.4, the simulation of a mechanism, seen at the very top level, is to deduce the behavior of the device based on descriptions of the structure and topology of the device, and information on external forces and energy as the input to the device. The detailed simulation algorithm is as follows:

Input: A network of nodes, with links representing the topology of the
device and nodes representing the component and connection nodes.
External force/energy input.

Simulator:

Given a component node, deduce its behavior according to its input;
if it's not a final node, propagate the motion-tendency description
to the next connection node; if it is a final node, either confirm
or deny the motion-tendencies proposed earlier by other nodes.

Given a connection node, deduce how the motion of the driving node
affects the motion of the follower node; propagate this causality
in terms of force/energy input to the next component node.

Selector:

The component node who has an external input is selected first.
After a component node is selected and worked upon, the connection
node which contains this component node is selected next.
After a connection node is selected and worked upon, the follower
node is selected next.

5.4 A Pair of Gears

5.4.1 Structure

The structure of a pair of gears consists of two identical gears and two identical shafts. The component nodes are as follows:

```
Node id: gear
Individuals:
   shaft
   gear
Properties:
   has-axle(gear, shaft)
   has-teeth(gear)
Precondition assumptions:
   held-against-gravity(gear, axle)
Physical variables:
   energy(gear)
   angular-velocity(gear)
   sign(angular-velocity(gear))
Component behavior description:
Method1:
   input: torque-applied-to(gear, DIR) > 0
   current state: angular-velocity(gear)=0 or
   next state: angular-velocity(gear) > 0,
               sign(angular-velocity(gear))=DIR
   output: rotating(gear, DIR) > 0

Method2:
   input:
   current state: angular-velocity(gear, DIR) > angular-velocity-min(gear, DIR)
   next state: angular-velocity(gear) > 0
   output: rotating(gear, DIR)

Method3:
   input:
   current state: angular-velocity(gear, DIR) < angular-velocity-min(gear, DIR)
   next state: angular-velocity(gear) = 0
   output: stationary(gear)

Method4:
   input: blocked(gear, SOMETHING, DIR)
   current state: angular-velocity(gear, DIR) > 0
   next state: angular-velocity(gear) = 0
   output: stationary(gear)
      or
   next state: angular-velocity(gear) > 0 or angular-velocity(gear) = 0
   output: break(SOMETHING)
```

```
        or
    next state: angular-velocity(gear) = unknown
    output: break(gear)

make-instance gear gear1, gear2

Node id: motor-driven-shaft
Individuals: shaft
Properties:
    shape(shaft, CYLINDRICAL)
Precondition assumptions:
    supported-against-gravity(shaft, SOMETHING)
    revolution-between(shaft, SOMETHING)
    motor-driven(shaft)
Physical variables:
Component behavior description:
Method1:
    input: torque-applied-to(shaft, DIR)
    current state:
    next state: angular-velocity(shaft, DIR) > 0
    output: rotating(shaft, DIR)

Method2:
    input: blocked(shaft, SOMETHING, DIR)
    current state: angular-velocity(shaft, DIR) > 0
    next state: angular-velocity(shaft, DIR) = 0
    output: blocked(shaft, SOMETHING, DIR)

make-instance motor-driven-shaft shaft1

Node id: shaft
Individuals: shaft
Properties:
    shape(shaft, CYLINDRICAL)
Precondition assumptions:
    supported-against-gravity(shaft, SOMETHING)
    rigidly-connected(shaft, SOMETHING)
Physical variables:
Component behavior description:
Method1:
    input: torque-applied-to(shaft) < torque-break(shaft)
```

```
   current state: angular-velocity(shaft) = 0
   next state: angular-velocity(shaft) = 0
   output: stationary(shaft)

   input: torque-applied-to(shaft) > torque-break(shaft)
   current state: angular-velocity(shaft) = 0
   next state: angular-velocity(shaft) = 0
   output: break(shaft)

make-instance shaft shaft2
```

5.4.2 Connection and Topology

The connection and topology of the gear pair consists of three connection nodes and has the following scheme:

```
Connection node id: shaft1-and-gear1
Individuals:
   gear-1
   shaft-1
Connection characteristics:
   supported-against-gravity(gear-1, shaft-1)
   rigidly-connected(gear-1, shaft-1)
Causal rules:
   input: rotating(shaft1, DIR) > 0
   output: torque-applied-to(gear1, DIR)

Connection node id: gear1-and-gear2
Individuals:
   gear-1
   gear-2
Connection characteristics:
   parallel(axis(gear1), axis(gear2))   % spur gear
   plane(gear1) = plane(gear2)          % aligned(gear1, gear2)
   tooth-meshed(gear1, gear2)
Causal rules:
Method1:
   input: angular-velocity(gear1, DIR)
   output: torque-applied-to(gear2, ~DIR)
```

```
Connection node id: gear2-and-shaft2
Individuals:
   gear2
   shaft2
Connection characteristics:
   held-against-gravity(gear2, shaft2)
   revolution-between(gear2, shaft2)
Causal rules:
Method1:
   input: angular-velocity(gear2, DIR) > 0
   output: nothing to shaft
```

5.4.3 Result of Simulation

The result of the simulation of the gear mechanism is shown in Figure 13.

5.5 Spring-driven Ratchet Mechanism and Intermittent Motion

5.5.1 Structure

Pictured in Figure 14 is a spring-driven ratchet mechanism. This device exhibits intermittent motion. The structure of this device consists of a spring, a cam, an arm, and a ratchet wheel. The component nodes for the device is as follows:

```
Node id: spring
Individuals:
   spring
   wall
Properties:
   elastic(spring)
   rigid(wall)
Precondition assumptions:
   held-against-gravity(spring, wall)
Physical variables:
   NOMAL= the length of the spring under no force influence
   length(spring)
   energy(spring)
```

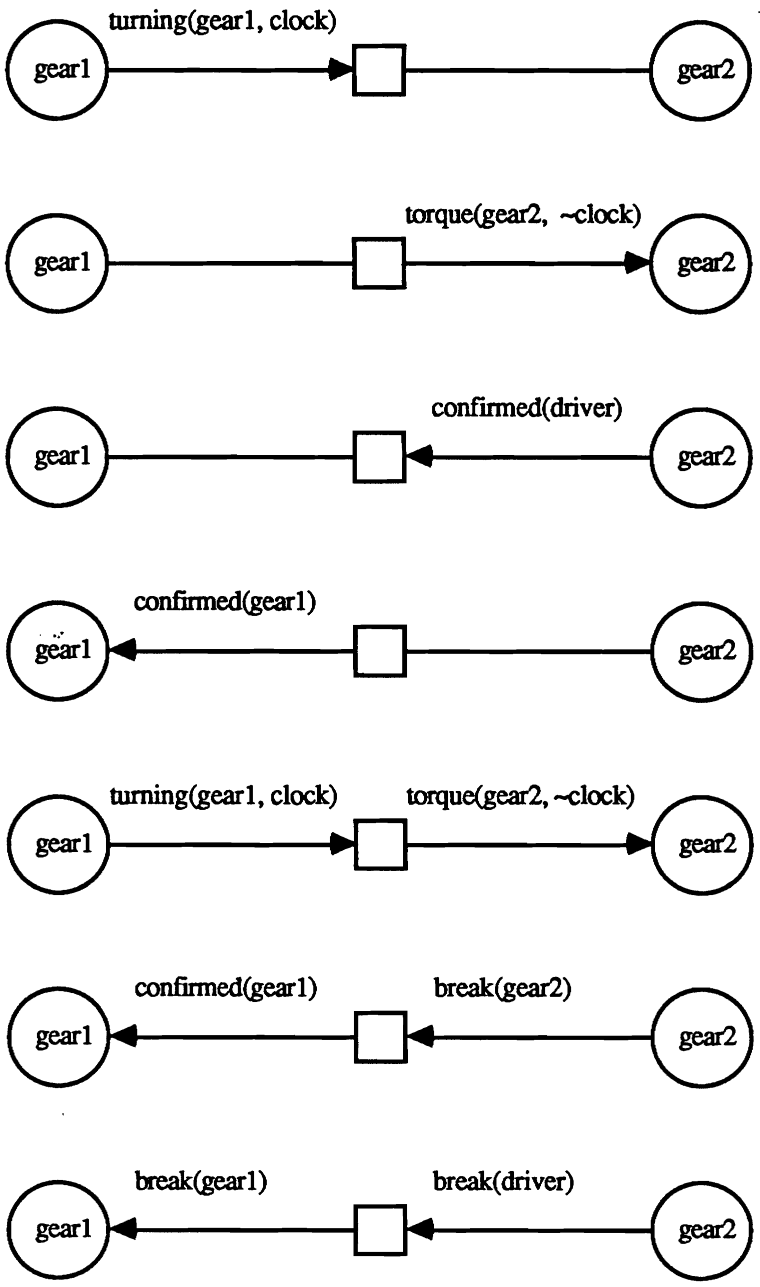

Figure 13: Causality description of the gear-pair motion

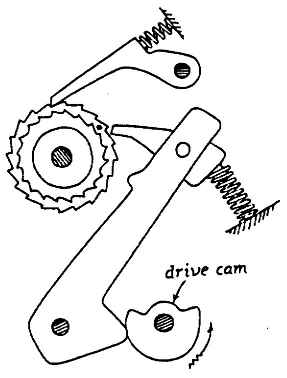

Figure 14: Spring-driven ratchet mechanism

```
Component behavior description:
Method1:
   input: force-applied-to(spring, X_NEG)
   current state: length(spring)= NORMAL, energy(spring)= 0
   next state: length(spring) > NORMAL
   output: suppressed(spring, X_NEG)

Method2:
   input: force-applied-to(spring, X_POS)
   current state: length(spring)= NORMAL, energy(spring)= 0
   next state: length(spring) > NORMAL
   output: pulled(spring, X_POS)

Method3:
   input: free(spring, X_NEG)
   current state: length(spring) > NORMAL, energy(spring) > 0
   next state: length(spring)= NORMAL
   output: released(spring)

make-instance spring spring-21

Node id: arm
Individuals:
   arm
   pivot
Properties:
   has-pivot(arm, pivot)
   rigid(arm)
```

```
Preconditions assumptions:
   held-against-gravity(arm, pivot)
   revolution-between(arm, pivot)
Physical variables:
   angular-velocity(arm)
Component behavioral description:
Method1:
   input: torque-applied-to(arm, dir), free(arm, dir)
   current state: angular-velocity(arm) = 0
   next state: angular-velocity(arm) > 0
   output: rotating(arm, dir)

Method2:
   input: free(arm, dir)
   current state: angular-velocity(arm) > 0
   next state: angular-velocity(arm) > 0
   output: rotating(arm, dir)

Node id: cam
Individuals:
   cam
   cam-shaft
Properties:
   rigid(cam)
   rigid(cam-shaft)
Preconditions:
   held-against-gravity(cam, cam-shaft)
   no-motion(cam, cam-shaft)
Physical variables:
   velocity(cam)
   position(cam)
Component behavior description:
Method1:
   input: torque-applied-to(cam, dir)
   current state: velocity(cam, dir)= 0
   next state: velocity(cam, dir) > 0
   output: rotating(cam, dir)

Method2:
   input:
   current state: velocity(cam, dir) > 0
   next stae: velocity(cam, dir) > 0
   output: rotating(cam, dir)

Node id: wheel
Individuals:
   wheel
   wheel-shaft
Properties:
   rigid(wheel)
   has-tooth(wheel)
   rigid(wheel-shaft)
Preconditions:
   held-against-gravity(spring, wall)
Physical variables:
   velocity(wheel)
   position(wheel)
Component behavior description:
Method1:
   input: force-applied-to(wheel, dir)
   current state: velocity(wheel)= 0
   next state: velocity(wheel, dir) > 0
   output: rotating(wheel, dir)

Method2:
   input:
   current state: velocity(wheel, dir) > 0
   next state: velocity(wheel, dir) > 0
   output: rotating(wheel, dir)

Method3:
   input: blocked(wheel, dir)
   current state: velocity(wheel, dir) > 0
   next state: velocity(wheel, dir) = 0
   output: stopped(wheel, dir)
```

5.5.2 Connection and Topology

The connection and topological description of the spring-driven cam mechanism consists of three connection nodes:

```
Connection node id: cam1-and-arm1
```

```
Individuals:
   cam1
   arm1
Connection characteristics:
   cam-type(cam1, arm1)
Causal rules:
Method1:
   input: rotating(cam1) contact(arm1, cam1)
   output: torque(arm1, CLOCK)

Method2:
   input: rotating(cam1), contact(arm1, cam1)= False
   output: nothing(arm1)

Connection node id: arm1-and-spring1
Individuals:
   arm1
   spring1
Connection characteristics:
   touch(arm1, spring1)
Causal rules:
   input: rotating(arm1, CLOCK)
   output: force-applied-to(spring1)

Connection node id: arm1-and-wheel1
Individuals:
   arm1
   wheel1
Connection characteristics:
   touch(arm1, wheel1)
Causal rules:
   input: rotating(arm1, CT-CLOCK)
   output: torque-applied-to(wheel, CT-CLOCK)
```

5.5.3 Result of Simulation

The result of the simulation of the spring-driven cam mechanism is shown in Figure 15.

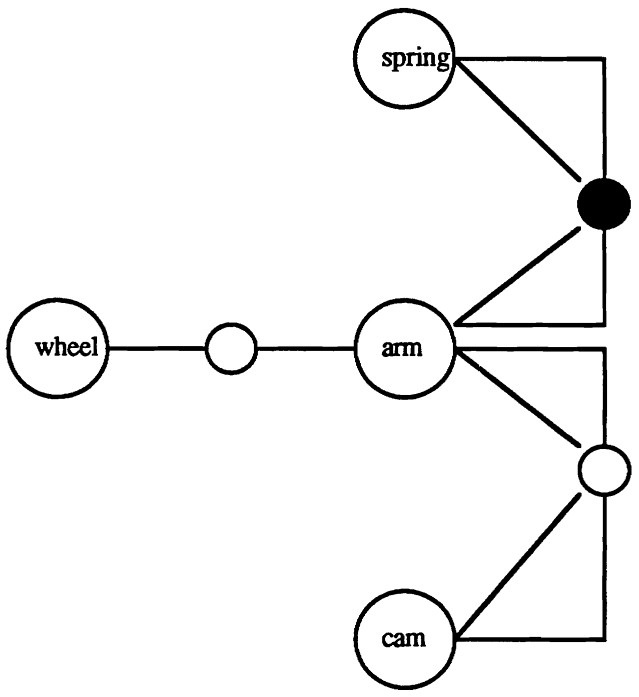

Figure 15: Causality description of the spring-cam mechanism

5.6 A Pendulum Clock

The pendulum clock example given here is the same clock from Section 3.4. It consists of four parts: a pendulum, an anchor, a wheel, and a weight. There are three connections: the connection between the pendulum and the anchor, the anchor and the wheel, and the wheel and the weight. The component and connection nodes for the pendulum clock is as follows:

5.6.1 Strucutre, Connection and Topology

```
Node id: pendulum
Individuals:
   hinge, rod, bob
Properties:
   weight(bob)> 0
   rigid(rod)
   has-hinge(pendulum, hinge)
Preconditions:
   suspended-against-gravity(rod, hinge)
   revolution-between(rod, hinge)
   rigidly-connected(bob, rod)
Physical variabels:
   position(bob)
   velocity(bob)
Behavioral descriptions:
   Method(swing-left):
      =>: none
      T1: position(bob)= MIDDLE
          velocity(bob, CLOCK-WISE) > 0
      T2: position(bob)= LEFT
          velocity(bob)= 0
      <=: swinging(pendulum, CLOCK-WISE)

   Method(swing-right):
      =>: none
      T1: position(bob)= MIDDLE
          velocity(bob, CT-CLOCK-WISE) > 0
          amount(velocity(bob)) > 0
          sign(velocity(bob))= CT-CLOCK-WISE
```

```
     T2: position(bob) = RIGHT
         velocity(bob) = 0
      <=: swinging(pendulum, CT-CLOCK-WISE)

  Method(swing-down-from-right):
     =>: none
     T1: position(bob)= RIGHT
         velocity(bob)= 0
     T2: position(bob)= MIDDLE
         amount(velocity(bob)) = MAX
         sign(velocity(bob))= CLOCK-WISE
     <=: swinging(pendulum, CLOCK-WISE)

  Method(swing-down-from-left):
     =>: none
     T1: position(bob)= LEFT
         velocity(bob)= 0
     T2: position(bob)= MIDDLE
         amount(velocity(bob)) = MAX
         sing(velocity(bob))= CT-CLOCK-WISE
     <=: swinging(pendulum, CT-CLOCK-WISE)

  Method(getting-energy):
     =>: kinetic-energy(bob) > 0   \* an initial kick by an agent *\
         sign(velocity(input-agent))= DIR
     T1: position(bob)= MIDDLE
         velocity(bob)= 0, sign(velocity(bob))=0
     T2: position(bob)= MIDDLE
         amount(velocity(bob)) > 0, sign(velocity(bob)= DIR
     <=: none

Node id: (pendulum, anchor)
Individuals:
   pendulum, anchor
Connection Characteristics:
   rigidly-connected(pendulum, anchor)
% the friction is big enough so that there is no motion between
% the pendulum and anchor
Causal rules:
   Method1:
      => swinging(pendulum, DIR)
      <= torque-applied-to(anchor, DIR)

Node id: anchor
Individuals:
   anchor
   hinge
Properties:
   has-hinge(anchor, hinge)
Preconditions:
   suspended-against-gravity(anchor, hinge)
   revolution-between(anchor, hinge)
Physical variables:
   position(anchor)
Behavior descriptions:
   Method(swing-left):
      =>: torque-applied-to(anchor, CLOCK-WISE) > 0
      T1: position(anchor)= MIDDLE
      T2: position(anchor)= LEFT
      <=: swinging(anchor, CLOCK-WISE)

   Method(swing-right):
      =>: torque-applied-to(anchor, CT-CLOCK-WISE) > 0
      T1: position(anchor)= MIDDLE
      T2: position(anchor)= RIGHT
      <=: swinging(anchor, CT-CLOCK-WISE)

   Method(swing-down-from-right):
      =>: torque-applied-to(anchor, CLOCK-WISE)
      T1: position(anchor)= RIGHT
      T2: position(anchor)= MIDDLE
      <=: swinging(anchor, CLOCK-WISE)

   Method(swing-down-from-left):
      =>: torque-applied-to(anchor, CT-CLOCK-WISE) > 0
      T1: position(anchor)= LEFT
      T2: position(anchor)= MIDDLE
      <=: swinging(anchor, CT-CLOCK-WISE)

Node id: (anchor, wheel)
Individuals:
   anchor
   wheel
Connection characteristic:
   blocked(wheel, left-arm(anchor),position(anchor)=RIGHT)
   blocked(wheel, right-arm(anchor), position(anchor)=LEFT)
   aligned(wheel, anchor)
Causal rules:
   Method1:
      =>: contact-detection(anchor, wheel)= blocked(wheel, left-arm(anchor))
      <=: blocked(wheel)

   Method2:
      =>: contact-detection(anchor, wheel)= no-obstacle(wheel,DIR)
      <=: free(wheel, DIR)

   Method3:
      =>: contact-detection(anchor, wheel)= blocked(wheel, right-arm(anchor))
      <=: blocked(wheel)

Node id: weight
individuals:
   weight-11
   string
   spindle
Properties:
   weight(weight-11) > 0
   flexible(string)
Preconditions:
   suspended-against-gravity(weight-11, string)
   wound-on(string, spindle, CT-CLOCK-WISE) % so that, the weight can gradually pull the string
Physical variables:
   height(weight-11)
Behavior descriptions:
   Method1:
      =>: gravity
      T1: height(weight-11)= H where H > 0
      T2: height(weight-11)= H + delta H
      <=: rotating(spindle, CLOCK-WISE)
% by making object a variable, we are ensuring the no-function-in-structure
% principle. Later when this message gets passed to (weight, wheel), object
% should be instantiated to wheel.
```

```
Node id: (weight, wheel)
Individuals:
   weight-11
   string
   spindle
   wheel
Connection characteristic:
   rigidly-connected(wheel, spindle)
Causal rules:
   Method1:
      =>: rotating(spindle, DIR)
      <=: torque-applied-to(wheel, DIR)

Node id: wheel
Individuals:
   wheel
   spindle
Properties:
   has-teeth(wheel)
Preconditions:
   held-against-gravity(wheel, spindle)
Physical variables:
   angle(wheel)
Behavior description:
   Method1:
      =>: torque-applied-to(wheel,DIR) > 0 and
          free(wheel, DIR)
      T1: angle(wheel)= theta
      T2: angle(wheel)= theta + delta(theta)
      <=: rotating(wheel, DIR)

   Method2:
      =>: torque-applied-to(wheel, DIR) and
          blocked(wheel)
      T1: angle(wheel)= theta
      T2: angle(wheel)= theta
      <=: no-motion(wheel)
```

5.6.2 Result of Simulation

Chapter 6

Toward Thesis Completion

6.1 What Has Been Done

6.1.1 Focusing the Research

We started our research with the goal of solving the repair automation problem. We defined the problem formally in Chapter 3. It was shown that this problem requires further research in many areas. We then decided to focus our attention on a crucial area in achieving repair, which is to capture the knowledge of how mechanisms work.

6.1.2 What Accounts for an Understanding of the Way Mechanisms Work

In Chapter 4, we presented a detailed discussion of what accounts for the knowledge of how mechanisms work. We discovered that when a person describes and explains how mechanisms work, he or she uses more "commonsense" and heuristic knowledge rather than mathematical and physical knowledge. Traditional science of mechanics, although precise in nature, has failed to provide such "commonsense" and heuristic knowledge.

6.1.3 A New Knowledge Representation Scheme: QUORUM

We examined existing knowledge representation schemes in Chapter 2. We discovered that none of the existing representations is adequate for the problem we were trying to solve. Thus, we constructed in Chapter 4 a new knowledge representation scheme, QUORUM, which aims at capturing the necessary knowledge defined earlier. QUORUM is able to provide:

1. causality reasoning;
2. behavioral descriptions for intermittent, continuous, reciprocal motions;
3. relation between geometry and motion;
4. relation between connection and motion propagation.

6.2 Towards Thesis Completion

To show that QUORUM is indeed sufficient to capture enough knowledge of how mechanisms work to do repair, we set the following goals for the next stage of research work:

1. To implement QUORUM and associated algorithms;
2. To implement prediction algorithm to deduce faulty behavior
3. To validate that the representation is adequate to simulate mechanisms at the qualitative level;
4. To demonstrate that the representation is able to produce behavioral predictions for such repair operations as the detaching of some parts in a device (generation of warning messages).

6.3 Future Tasks

Possible suggestions concerning future tasks are summarized as follows:

1. Integrating QUORUM with quantitative systems;
2. Building user interfaces;
3. Constructing reasoning systems that handle disassembly planning with regard to warning messages;

Appendix A

Semantics of QUORUM Language

A.1 Predicates

aligned(object1, object2) This denotes the condition when two objects are horizontally aligned so that the two objects are physically in contact with each other and furthermore the axles are parallel to each other. For example, aligned(gear1, gear2) makes possible for motion propagation between two gears.

blocked(object1, object2, direction) Object1 is blocked by object2 so that object1 is not free to move in direction **direction**.

break(object) This predicate denotes the fact that the geometrical property of the object is changed. In particular, when an object is broken, often it is broken into more than one piece.

flexible(obj) The object is made of flexible material. For instance, a string, a thin wire are flexible objects. Force can be transferred through a flexible object only by pulling.

flexible-tied-together(obj1, obj2) Two objects are tied together. Object2, however, can move with at most 3 degree of rotational freedom with respect to object1.

force-applied-at(obj, place) In dealing with force and motion, it is important to know where the force is applied. I have not figure out the representation for place yet. That is I don't know whether it should be coordinates or a qualitative symbol like Forbus' notion.

free(obj, dir) This denotes the degree of freedom in direction **dir**. In stead of saying that an object has three degree of freedom along x, y, z axis, we have to say free(obj, x-axis), free(obj, y-axis) and free(obj, z-axis). We are trading uniformity for tediousness.

has-axle(object) For an object to rotate, there must exists a rod or a shaft supporting that object and serving as an axle.

has-hinge(object, hinge-object) Object has a hinge denoted by *hinge-object*. For instance, **has-hinge(anchor, collar)** denotes the fact that an anchor has a collar so that it can turn on the collar.

has-pivot(object)

has-teeth(object)

held-against-gravity(object1, object2) Object1 is held from falling down by object2. If such relation fails to hold, gravity will take effect and object1 fall down.

is-a(something, class) An object, or a relation, is a kind of a class of objects, or relations.

motion(obj, type) The object is exhibiting a certain type of motion. The type of motion an object exhibits depend on the characteristics of force and the nature of the object. We will see exactly how motion types are determined when later we discuss the various rules for motion type determination.

parallel(dir1, dir2) The two directions are parallel to each other.

part-of(submodule, module) For instance, a pendulum bob is part of a pendulum. Thus we write: part-of(bob, pendulum).

perpendicular(dir1, dir2) The two directions are perpendicular to each other.

revolution-between(object1, object2, ..., hinge-object) All of these objects are threaded together by a type of hinge. The friction between the shared surfaces of these objects is small enough so that each object is allowed to turn around the hinge. There is no sliding motion allowed for any of these objects along the hinge.

rigid(obj) The object is made of rigid material,meaning force can be transferred either by pulling or pushing.

rigidly-connected(object1, object2, ..., joint-object) All of these objects are jointed rigidly together by a joint, usually a discrete joint, such as screw, rivet, etc. The condition is true when all of these objects can be treated as one body. This relationship will become false when the geometrical constrainst imposed on these two objects falls apart, such as a screw gets loosened.

rigidly-connected(object1, object2, ...) These objects are connected to each other without a joint. This is what is called integral joint, meaning the joining of objects by themselves. There should be no motion allowed between the objects.

rotating(object, dir) Object is rotating around an axis denoted by *dir*.

shape(obj, shape-type) Denotes whether the shape of the object is square, ractangular, round, elliptical.

sliding(object1, object2, ..., shaft-object) All of these objects are connected together by a type of shaft. All the objects are allowed to slide along the shaft. Turning is not possible.

stationary(object) An object is not in motion. This condition is to be distinguished from *blocked*. A stationary object is subject to motion if force is applied properly.

supported-against-gravity(object, surface) The object is laid on a surface. Depending on the angle between the surface and the x-axis, the object can slide or roll down if the slope of the surface is steep enough.

suspended-against-gravity(obj, something) The object is fastened from above by a suspensor, e.g., a string, a rod, etc., so as to allow free movement at the point of suspension.

swinging(object, dir) Object is swinging in the direction denoted by *dir*. Thus, **swinging(pendulum, x-axis)**, and **swinging(pendulum, dir(x,y,z))** denote the motion swinging (or oscillating) of a pendulum in different directions.

tooth-meshed(object1, object2) This denotes the geometrical conditions of two toothed objects. The geometry of those two objects are so designed that when they are put against each other, they interlock each other.

torque-applied-to(object, dir) There is a certain amount of torque applied in direction *dir* to the object so that the object is subject to rotation in that direction.

wound-on(object1, object2, dir) *Object1* must be a flexible object. *Object1* is wound onto *object2* so that when this mechanism is unwinding, *object2* is rotating in the opposite direction of *dir*.

A.2 Functions

We use quantities to describe an object. For instance, the position, velocity, and external force can all be used to describe an object. For certain quantities, such as velocity, we also need to know the sign, to be distinguished from the magnitude, of that quantity. Therefore we use *amount* and *sign* to separate the two parts of a quantity, if necessary.

Quantities correspond to the numerical-valued functions in predicate calculus. A function denotes a term or an individual. A numerical-valued function denotes a numerical value. For instance, **position(object)** denotes the position of an object. A function can be further used in another function. For instance, **sign(velocity(object))** denotes the sign of the velocity of an object.

acceleration(object)

amount(quantity)

angle(object,frame)

angle-(object, x-axis, alpha) Alpha denotes the angle that object forms with the x-axis. For instance, angle(string, x-axis, 45) says that a string, probably used to hang a block, forms an angle of 45 degrees with the x-axis.

axis(object) A real and imaginary straight line passing through the object that actually or supposedly revolves upon it.

dir(x,y,z)

force(object)

height(object)

kinetic-energy(object)

left-part(object)

plane(object) The plane the object lies in.

position(object,frame)

potential-energy(object)

right-part(object)

sign(quantity)

velocity(object

weight(object)

Bibliography

[1] Daniel B. Dallars, editor. *Tool and Manufacture Engineers Handbook.* McGraw-Hill, 1976.

[2] Johan de Kleer. Multiple representations of knowledge in a mechanics problem-solver. In *Proceedings Fifth International Joint Conference on Artificial Intelligence*, August 1977.

[3] Johan de Kleer. A qualitative physics based on confluences. *Artificial Intelligence*, 24, 1984.

[4] Johan deKleer and John Seely Brown. Mental models of physical mechanisms and their acquisition. In *Cognitive Skills and Their Acquisition*, J.R. Anderson, 1981.

[5] Johan deKleer and John Seely Brown. The origin, form and logic of qualitative physical laws. In *Proceedings Eighth International Joint Conference on Artificial Intelligence*, August 1983.

[6] Boi Faltings. *a theory of qualitative kinematics in mechanisms.* Technical Report uiucdcs-r-86-1274, university of illinois at urbana-champaign, 1986.

[7] Paul A. Fishwick. *Hierarchical Reasoning: Simulating Complex Processes over Multiple Levels of Abstraction.* Technical Report, Computer Science Department, University of Pennsylvania, 1986.

[8] F.Reuleaux. *The Kinematics of Machinery.* Macmillan and Co., 1876.

[9] Andrew Gelsey. Automated reasoning about machine geometry and kinematics. In *Third IEEE conference on artificial intelligence applications*, February 1987.

[10] Higgins and Morrow, editors. *Maintenance Engineering Handbook.* 1977.

[11] K.Forbus. Qualitative process theory. *Artificial Intelligence*, 24, 1984.

[12] Benjamin Kuipers. Commonsense reasoning about causality: deriving behavior from structure. *Artificial Intelligence*, 24, 1984.

[13] Burton Paul. *Kinematics and Dynamics of Planar Machinery.* Prentice Hall, 1979.

[14] Stephen Michael Platt. *A Structural Model of The Human Face.* Technical Report, Computer Science Department, University of Pennsylvania, 1985.

[15] Chuck Rieger and Milt Grinberg. The declarative representation and procedural simulation of causality in physical mechanisms. In *Proceedings Fifth International Joint Conference on Artificial Intelligence*, August 1977.

[16] Peter Schwamb, Allyne L. Merrill, and Walter H. James. *Elements of Mechanism.* John Wiley & Sons, Inc., 1947.

[17] Simon and Schuster, editors. *The Way Things Work.*

[18] Craig Stanfill. *Form and Function: The Representation of Machines.* Technical Report TR-1347, Computer Science Department, University of Maryland, 1983.