

From Diagrams to Models by Analogical Transfer

Patrick W. Yaner and Ashok K. Goel

Design Intelligence Group
Artificial Intelligence Laboratory
Division of Interactive and Intelligent Computing
College of Computing, Georgia Institute of Technology
Atlanta, GA 30332-0280
{yaner, goel}@cc.gatech.edu

Abstract. We present a method for constructing a teleological model of a drawing of a physical device through analogical transfer of the teleological model of the same device in an almost identical drawing. A source case, in this method, contains both a 2-D vector-graphics line drawing of a physical device and a teleological model of the device called a Drawing-Shape-Structure-Behavior-Function (DSSBF) model that relates shapes and spatial relations in the drawing to specifications of the structure, behavior and function of the device. Given an almost identical target 2-D vector-graphics line drawing as input, we describe how an agent may align the two drawings, and transfer the relevant structural, behavioral and functional elements over to the target drawing. We also describe how the DSSBF model of the source drawing guides the alignment of the two drawings. The Archytas system implements this method in domain of kinematic devices that convert translational motion into rotational motion, such as a piston and crankshaft device.

1 Background, Motivation and Goals

Diagram understanding is a persistent and central issue in research on diagrams. Novak [20] explicitly viewed the task of understanding a diagram as one of constructing a model for it: his Beatrix program understood a textually-annotated schematic diagram kinematics problem by constructing a structural model. In cognitive science, Narayanan, Suwa and Motoda [19] describe a psychological study in which human subjects built behavioral models of a physical device from its labeled schematic diagram. The computational advantage of model construction is that the model enables inferences for higher-level tasks, e.g., analysis of the functions of the physical device in case of Narayanan, Suwa, and Motoda's human subjects. The type of the output model in the task of diagram understanding depends in part on the input diagram and partly on the inferences desired of the model.

Current AI methods for constructing models from diagrams, e.g., in the Beatrix and the GeoRep [9] programs, often apply domain-specific rules to extract the model from the diagrammatic representation. For example, GeoRep first uses a domain-independent diagrammatic reasoner that extracts shapes and spatial relations from a given diagram, and then applies domain-specific rules for extracting a behavioral model from the intermediate representation of shapes and spatial relations. Given the state of the art in

In D. Barker-Plummer, R. Cox, & N. Swoboda (Eds.): *Diagrams 2006*, LNAI 4045, pp. 55–69, 2006. Springer.

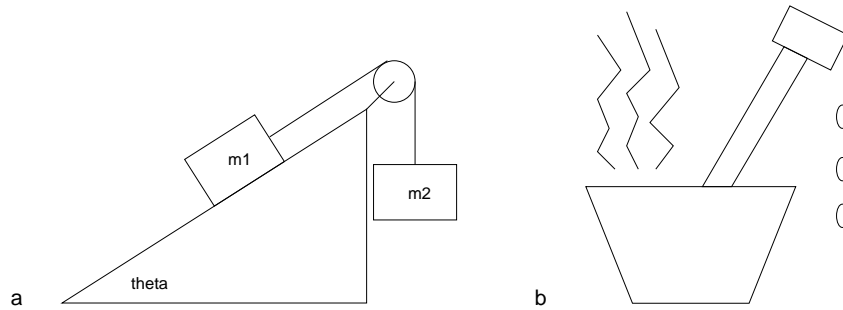


Fig. 1. (a) a sample input to Beatrix system (adapted from Novak [20]): a drawing of two block masses suspended over an inclined plane by a pulley; and (b) a sample input to the GeoRep system (adapted from Ferguson and Forbus [9]): a drawing of a cup of hot liquid with a metal bar and a melting ice cube at the other end of the metal bar.

research on diagrams in general, current AI programs typically focus on very simple diagrams. Figure 1(a) illustrates a sample diagram input to Beatrix; figure 1(b) similarly illustrates a sample diagram input to GeoRep.

Now let us consider the 2-D vector-graphics line drawing illustrated in figure 2(b). It depicts a piston and crankshaft device that converts linear motion of the piston (shown on the right side of the figure) into rotational motion of the crankshaft (shown on the left side; the axis of rotation is perpendicular to the page). Note however that drawing itself does not contain any labels or annotations. Given this input drawing, let us consider the task of constructing a teleological model of the device depicted in the drawing, where the teleological model specifies both the output functions of the device (what does it do?) and the internal causal behaviors that accomplish the functions (how does it work?). Understanding this diagram by constructing a teleological model that can answer the above questions is a complex task because the functions and the behaviors of the device are not directly apparent from the drawing.

But let us also suppose that the agent is familiar with this domain, and has come across many drawings and their teleological models in the domain. In particular, let us suppose that the agent already knows of the teleological model for the drawing in figure 2(a). Note that this source drawing is almost identical to the target drawing in figure 2(b); in fact, the two drawings depict the same piston and crankshaft device in two different states. Our first hypothesis in this work is that, under these knowledge conditions, the agent may generate a teleological model for the target drawing by *analogical transfer* of the teleological model of the source drawing.

In general, analogical reasoning involves the steps of retrieval, mapping, transfer, evaluation, and storage, as illustrated in figure 3. In this paper, however, we focus only on the tasks of mapping and transfer. The design domain is that of kinematics devices, and, in particular, devices that convert linear motion into rotational motion (and vice versa). Archytas is a computer program that implements our theory of diagram understanding by analogical reasoning. The source and target drawings are created using XFig (see www.xfig.org/), a vector-graphics program. Archytas begins its processing

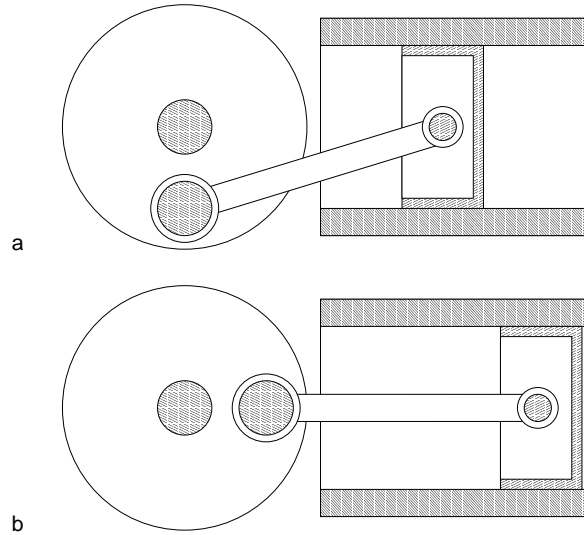


Fig. 2. (a) A sample source drawing of a piston and crankshaft assembly. The components depicted are the cylinder, the piston, the crankshaft, and the connecting rod. (b) A target drawing; the same device, with the piston now at the top of its range of motion (“top dead center”).

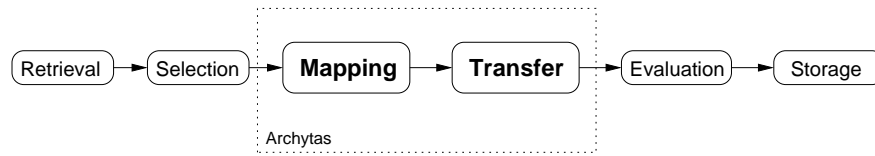


Fig. 3. The five major subtasks of analogy: retrieval, selection, mapping, transfer, and evaluation. This paper focuses on the mapping and transfer tasks implemented in the Archytas system.

of the target drawing with a representation of shapes and their locations generated by XFig.

Both mapping and transfer are complex tasks. The mapping task is complex because a given shape (or spatial relation) in the input target drawing may map into many similar shapes (or spatial relations) in the known source drawing. The transfer task is complex because based on a mapping between the shapes and spatial relations in the two drawings, the goal is to transfer the functions and behaviors of the device in the source drawing to the target drawing. This implies that we need some representation that relates shapes and spatial relations in a source drawing to the functions and behaviors of the device depicted in the drawing. In earlier work on adaptive design [10, 11, 12], we developed Structure-Behavior-Function (SBF) models of physical devices: an SBF model of a device uses the internal causal behaviors of the device as intermediate abstractions that relate the functions of device to its structure. Our second hypothesis in the present work is that the schema and ontology of SBF models can be productively expanded and extended into Drawing-Shape-Structure-Behavior-Function

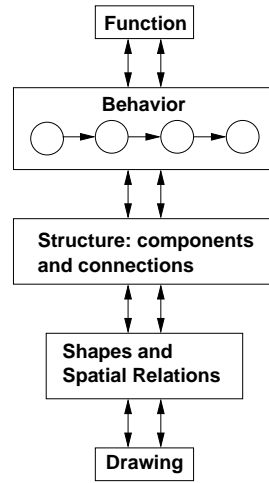


Fig. 4. The overall schema of a Drawing-Shape-Structure-Behavior-Function (DSSBF) model. This diagram shows the overall hierarchy, from function at the top, through the causal behavioral model, the structural components and connections, the shapes and spatial relations depicting them, and finally the original drawing.

(DSSBF) models: a DSSBF model uses structure as an intermediate abstraction to relate shapes and spatial relations in a drawing to the behaviors and functions of the device depicted in the drawing.

Given the complexity of the tasks of analogical mapping and transfer, our work on the Archytas project so far has focused on mapping and transfer between two nearly identical drawings. Specifically, in Archytas, we assume that the target and the source drawings are so similar that any distinction between them at the shape level makes no difference at the structural level of abstraction. Note that while this assumption of near identity of the source and the target drawings simplifies the transfer task, it makes little difference to the mapping task since the mapping occurs at the level of shapes and spatial relations. This implies that we need a method to control the complexity of mapping at the shape level. Our third hypothesis is that knowledge of the functions of the important shapes in the DSSBF model of the source drawing informs the mapping task at the shape level, and seeds the mapping with shapes that play a critical role in the functioning of the device.

2 DSSBF Teleological Models

As figure 4 illustrates, a DSSBF model of a physical device unifies the functional and spatial representations of the device. This unification results in a five-level model with function at the top and form (e.g., a drawing) at the bottom, with shape, structure and behavior as intermediate levels of abstraction. Note that in general there may be many drawings, and, hence, many shape representations of a single device (but figure 4 does not depict this for clarity).

Table 1. Outline of the structural model of the piston/crankshaft and example.

Component	Properties	Variable Quantities	Connected to
Piston	height, diameter	linear momentum	cylinder, connecting rod
Crankshaft	diameter	angular momentum	crankcase, connecting rod
Connecting Rod	length	angular, linear momentum	crankshaft, piston
Cylinder	diameter, length		piston, crankcase
Crankcase			cylinder, crankshaft

The representations of any two consecutive levels in the five-level DSSBF model contain two-way pointers to each other. For example, as in SBF models, the specification of a function specifies the behavior that accomplishes it and the specification of a behavior specifies the function (if any) that it accomplishes. Similarly, the specification of a behavior specifies the structural constraints (in the form of connections among components) that enable it, and the specification of a component specifies its functional abstractions and the role they play in a behavior. In addition, in a DSSBF model, the specification of a structural component or connection specifies the shape that depicts it in a drawing, and the specification of a shape specifies the component or connection it depicts. Thus, the organization of a DSSBF model of a physical device affords navigation of the entire model, and accessing of knowledge at one level of abstraction that is relevant to reasoning at another level.

2.1 An Illustrative Example

Figure 2 illustrates a piston and crankshaft device. In this device, there are five components, though only four are depicted in the figure: the (1) piston, (2) crankshaft, (3) connecting rod, (4) cylinder, and (5) crankcase (not shown). The function of this device is to turn the crankshaft. In the DSSBF language, this function is represented as a schema that specifies the state it takes as input, the state it gives as output, and the behavior that accomplishes the function.

In the DSSBF language, a behavior refers to an internal causal process, and is represented as a sequence of discrete states and state-transitions. The states in a behavior specify values of variables relevant to the behavior. The specification of the behavior of the crankshaft, for example, tracks the angular momentum of the crankshaft, which it gains from a downward force coming from the connecting rod through the joint with the connecting rod, and loses through friction. The annotations on a state transition specify the causes and the conditions of the transition.

In a DSSBF model, structure refers to the components and the connections among the components. Table 1 shows an outline of the specification of the components in the piston and crankshaft device. Briefly, each component schema specifies its properties, which take values, and quantities, which have a type of either scalar or vector, and which are variables whose values are changed by the causal processes in the behaviors of the device.

Figure 5 illustrates the connections in the piston and crankshaft example. In the DSSBF language, connections are represented as schemas. Connections also have types

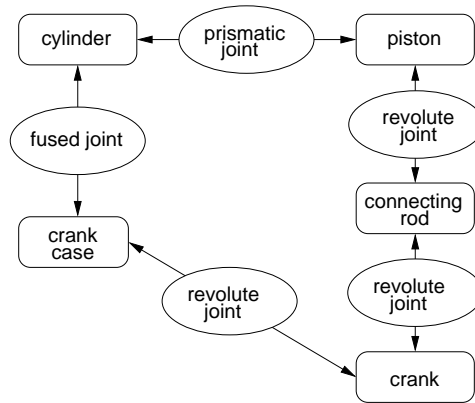


Fig. 5. A diagram of the connections in the structural model shown in table 1, where rounded boxes represent components and ovals represent connections between components. Properties and variable quantities of components are detailed in slot values for each component schema.

indicating their degrees of freedom, e.g., revolute has one degree of freedom (rotation), prismatic has one degree of freedom (translation), and so on.

Shapes and Spatial Relations: Let us consider the shape-level representation of the drawing of the source case illustrated in figure 2. Since this is XFig file, the properties and locations of lines and their interconnections already are known. Thus, Archytas begins with whole shapes, such as rectangles and circles, and their geometric properties. In the DSSBF language, each shape schema has slots associated with it detailing specific dimensions and aspects of the shape, such as the height and width of rectangles and the diameter of circles.

But Archytas needs to infer the relevant interrelationships among the shapes. For XFig drawings, the DSSBF language uses a taxonomy of spatial relationships among the shapes in a drawing: parallel-ness and perpendicularity, end-to-end and overlapping connections between lines, collinearity, horizontal and vertical alignment and relative length, and containment. Figure 6 illustrates a partial specification of the spatial relations in the piston and crankshaft drawing. This specification is an abbreviation of the actual representation: for instance, the component lines of the rectangles and the part-whole relations with the rectangles, as well as the interrelationships between them, are not shown, and only some of the spatial relationships are shown. The complete description, including mereological (part-whole) relations of component lines to polygons, involves hundreds of terms: the actual relational description of this drawing has 644 relations, including all relations involving individual lines that are parts of polygons. Excluding these component lines and their interrelations, the description had 72 relations in it.

Relating Shape to Structure: In order to be useful, the representation of the shapes and the spatial structure of the drawing must enter into a relationship with the structural

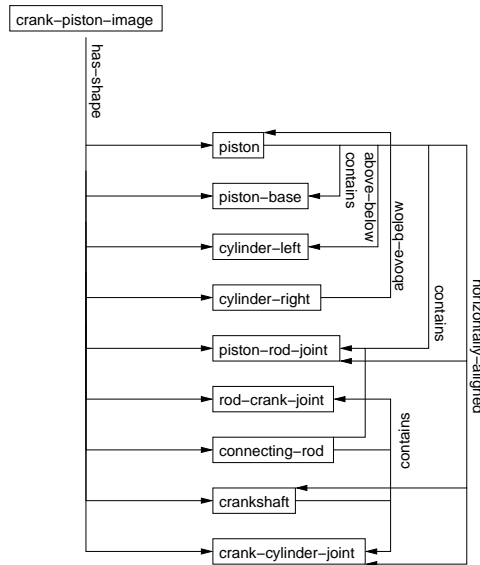


Fig. 6. Representation of most of the the shapes and only some of the principal spatial relations from figure 2(a). For clarity the labels on the shapes reflect the names of components depicted. This diagram shows only the properties and relations of the aggregate shapes, not the aggregation of lines into rectangles, or the interrelationships of these lines.

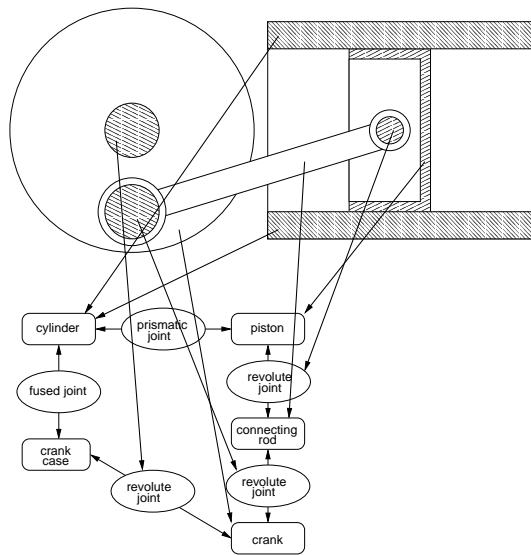


Fig. 7. The relation of shape to structure, showing the mapping between the shapes and the structure. The relations between the shapes and the components and connections are DEPICTS relations: *shape* DEPICTS *component*. In this figure, the drawing from figure 2(a) is shown for clarity, but in the actual representation the relations are between the shape nodes shown in figure 6 and the components and connections shown in 5.

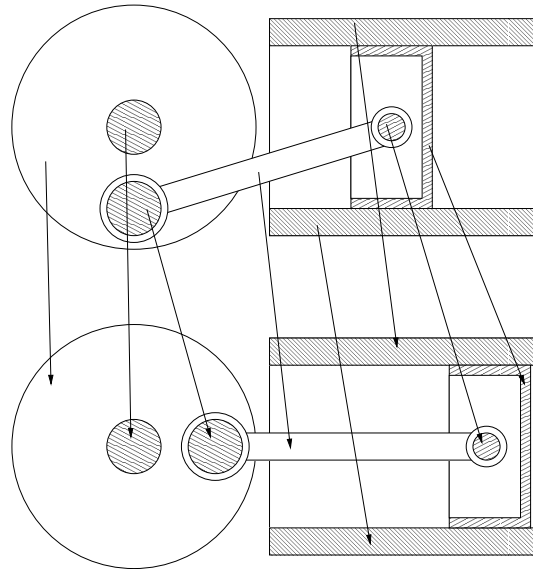


Fig. 8. The mapping between the images in figures 2(a) and 2(b). The actual mapping is between the representations of the sort shown in figure 6. Not all relations match: some relations in the source (on the top) will not be found in the target (on the bottom) and vice versa. This is only a single (best) mapping; the several smaller partial mappings are not shown.

elements in the model. These relations take the form of links between the shape schemas and the component schemas in the model. At the most basic level, we need relations of the form A DEPICTS B from the shape schemas to the components schemas, A being a shape schema, B being a component or connection schema. It is important to note that only the shapes themselves enter into these relationships; the relations between them do not.

3 Analogical Mapping

We have developed an algorithm for generating all partial mappings between given source and target drawings. (A “map” is a correspondence between individual entities or relations, and a “mapping” as a mutually consistent set of maps. A partial mapping is a mapping in which some relations do not necessarily match.) This algorithm gathers up individual maps from shapes in the source to shapes in the target drawing, and attempts to merge them into whole mappings from the source to the target drawing. Treating the shape and spatial relation representation as a labelled graph, this corresponds to the problem of maximal overlap set [4], also known as maximal common edge subgraph [21].

The algorithm begins with the source shapes and spatial relations and a target shapes and spatial relations. The algorithm marks some of the relations in the source representation as “important” so that no mapping is returned that does not involve at least one

of these relations. This drastically reduces the search space. Specifically, the algorithm uses the DSSBF model to determine which components are directly involved in behaviors of the device, and determine which shapes depict those components. Certain individual relations involving these particular shapes are marked as “important”.

In addition to this, there is a parameter to the algorithm that is a lower bound for the size of any mapping generated, so that Archytas will not return any mapping that is smaller than this bound (in terms of the number of relations mapped). There can be dozens or even hundreds of these small degenerate mappings of just one or two shapes, even when there’s a single complete mapping.

The procedure runs as follows:

1. Gather up all maps between source and target relations. Each map between a pair of relations will entail two maps between the entities related, so that, if A contains B maps to X contains Y , then this means A maps to X and B maps to Y
2. Pull out those maps involving marked (“important”) relations as “important” maps.
3. Choose one of the “important” term maps m_1 . Now, recursively gather every other map m_i , for $i > 1$, such that m_1 and m_i are consistent. They are consistent when:
 - the source terms in m_1 and m_i are different
 - the target terms in m_1 and m_i are different
 - the associated entity maps are consistent (same source entity maps to the same target entity, and conversely the same target entity has the same source entity being mapped to it)
 These rules enforce a one-to-one mapping between both relations and entities.
4. When all of the mutually consistent individual maps have been gathered, save this as a single (partial) mapping if it exceeds the minimum size for an acceptable mapping (minimum bound).
5. Choose the next marked “important” term map m_2 , and repeat steps 3 and 4, iterating through all of the marked “important” term maps.
6. Return the resulting list of mappings.

The third step—expanding the mapping from the “important” map—involves a recursive operator that expands a given mapping with a set of candidates and excluding those in another set, initially null. This procedure (not shown) is based on an algorithm of Bron and Kerbosh [1].

These mappings allow term-by-term comparison of the source and target, so that Archytas can reason about the similarities and differences between the two drawings with respect to a potential alignment of the drawings. Note that each individual mapping is a term mapping (associated relations), as described above, and must be converted into mappings between entities. The method returns all *maximal* complete mappings, and as such the largest of these will be the *maximum*; a *maximal* mapping is one that cannot be expanded to include other terms without contradiction, and a *maximum* mapping is the largest of all of the maximal mappings. Thus, if there is a subgraph isomorphism between source and target either way (source entirely aligns with target, or vice versa), the algorithm will return no partial mappings that are consistent with that particular mapping, but it may return others that are inconsistent with it.

In the context of figures 2(a) and 2(b), all the shapes from the source map onto shapes from the target, but the algorithm does discover that the target (figure 2(b)) has

the connecting rod rectangle parallel with the cylinder and the piston, but the source (figure 2(a)) does not. It also returns several partial mappings: for instance, the top rectangle for the cylinder in figure 2(a) may map to the bottom rectangle for the cylinder in figure 2(b) or the top one. These are inconsistent with each other, but both would be maximal, and so the algorithm returns them both.

3.1 Transfer

Once we have a mapping between the source and target shapes, we need to transfer the teleological model. Since we have assumed that the source and target drawings are so similar that any distinction between them makes no difference at the structural level, the transfer task is straightforward. The basic problem is that Archytas has mappings between shapes in each drawing, and from those mappings Archytas can hypothesize that they should therefore depict the same component or connection. Thus, the procedure is to begin with the mapped shapes, and transfer the components and connections depicted by those mapped shapes, reconstructing the model iteratively. This process is shown in figure 9. Here is an outline of the basic algorithm:

1. Transfer `DEPICTS` relations, setting up relations to new components for each shape that depicts some component.
2. Transfer motion constraints (e.g. piston moves horizontally within the rectangles representing the cylinder).
3. From these, transfer the structural frames themselves:
 - (a) Transfer depicted components, their properties and variable quantities
 - (b) Transfer depicted connections between components (e.g. the piston/rod joint and the rod/cylinder joint, both represented explicitly by circles).
 - (c) Transfer non-depicted connections amongst depicted components (e.g. piston/cylinder joint, which is merely a spatial relation).
 - (d) Hypothesize new non-depicted components for each non-depicted component in the source (e.g. crankcase).
 - (e) For each component that maps onto a new hypothesized target component (depicted or undepicted), transfer all of the parameters, quantities, and primitive functions from the source to the target for that component.
 - (f) For each component that is involved in a behavior, transfer that behavior, iterating through each state and transition one by one.
 - (g) Some states are linked to the functional specification of the design, as are the behaviors of which they are part; for each behavior that realizes a function (“function F by behavior B”), transfer that functional specification from source to target.

Using this method we can hypothesize that each of the components depicted in figure 2(a) is also depicted in figure 2(b), and that the essential spatial constraints are not violated, and that therefore the undepicted component (the crankcase) and connections (those involving the crankcase) must be part of the target model as well.

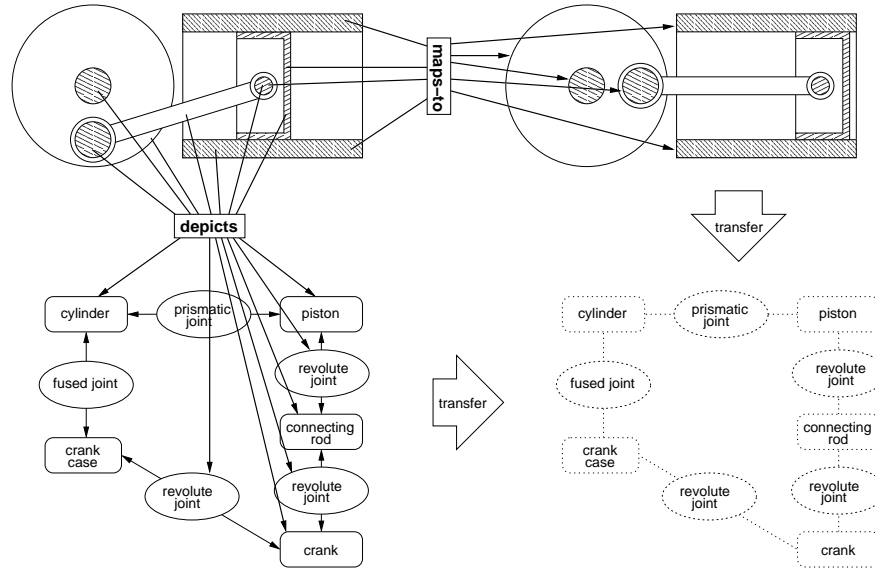


Fig. 9. The process of analogical transfer. For clarity, the mappings shown are partial. Given a mapping from the source (on the left) to the target (on the right), and a structural model for the source (bottom left), Archytas constructs new hypothetical structural model for the target (bottom right) by analogical transfer.

4 Related Work

This research lies at the intersection of several lines of investigation in the cognitive sciences, including diagrammatic reasoning, analogical reasoning, qualitative models, and model construction. Here we relate this work only with the most similar AI work on diagram understanding by model construction, and analogical mapping with spatial representations.

Much earlier work on constructing models from drawings used forward-chaining rule-based reasoning. The early Beatrix system [20] for example, used domain-specific rules to construct a structural model of simple kinematics devices (e.g., a block on an inclined plane—see figure 1(a)) from a textually-annotated 2D diagram.

The more recent GeoRep system [9] takes as input simple a 2D vector-graphics line drawing depicting a physical process, e.g., a cup with steam coming out of it (figure 1(b)). It gives as output a symbolic description of the physical process depicted in the drawing, e.g., steam coming out of hot liquid contained in the cup. GeoRep is organized as a two-stage forward-chaining reasoner. First, a low-level, domain-independent relational describer recognizes shapes and spatial relations in the input line drawing, and then a high-level domain-specific relational describer applies domain-specific rules to produce an final description of the physical process in the diagram.

Joskowicz has done work on deriving behavioral descriptions from geometrical and topological descriptions of the parts of a device [16]. His work focussed on deriving

constraints on possible motion in so-called “kinematic pairs” in a “Local Interactions Analysis”, and then using constraint propagation to derive a qualitative “functional” description of the device in a “Global Interactions Analysis” (despite the use of the term “functional description”, Joskowicz is speaking of what we are calling behavior). In more recent work, Dar, Joskowicz, and Rivlin [5] have derived similar descriptions from digital image sequences.

The SketchIT system [23] takes as input a 2D sketch of a physical device, and gives as output multiple designs of the physical device in the kinematics domain, where each design is augmented with a simple state-transition diagram to describe the device behavior. The system first produces “a behavior-ensuring parametric model” (or BEP model) of the components of the design, and from this determines geometric constraints on the motion of the parts, generating all qualitative configuration spaces consistent with the behavioral constraints. Next, it selects motion types for each component, and, finally, from the motion types and the geometric interpretations provided by a library of interaction types, it generates a BEP model for the design as a whole. In contrast to all these earlier AI projects on diagram understanding by model construction, our work uses analogical reasoning for constructing models from diagrams.

Evans’ early ANALOGY program [6] solved four-term geometric analogies of the $A : B :: C : ??$ sort one might see on intelligence tests, where the task is to choose from several given alternatives the geometric figure that best completes the analogy. Given a multiple choice question of this kind, ANALOGY attempted to find a procedure for turning A into B, and then turned C into each of the (say) 5 answers, and whichever of those transformation procedures, represented as a labelled graph, was closest to the original was chosen. It used a simple language of primitives (dots, lines, closed polygons, etc.), relationships (above, left-of, and so on), and transformations (rotate, reflect, expand, contract, add, delete) for this task.

The more recent LetterSpirit program [17, 15] takes a stylized seed letter as input and outputs an entire font that has the same style (or “spirit”). The system understands “roles”, such as the crossbar of an f or a t. It makes new fonts by determining some attributes such as “role traits” (e.g. crossbar suppressed), “motifs” (geometric shapes, such as a parallelogram, used over and over again), and “abstract rules” (e.g. no diagonal lines, only horizontal and vertical), and using these attributes builds an entire alphabet in some new style.

The Structure-Mapping-Engine (SME) [7] is a powerful, but content-free, analogical mapping system. SME first generates local maps between the target and the source graphs, then uses heuristics based purely on the structure of the graphs to select among the local maps, and finally to build a consistent mapping. JUXTA [8] uses SME to compare two nearly identical drawings of a physical process, such as two drawings of a coffee cup with a metal spoon, but with a thicker metal bar in one drawing than in the other. JUXTA first uses GeoRep for deriving structure from shape, and then uses SME to compare the two structures, looking for alignable differences (that is, differences in associated attribute values in two schemas), and based on these differences drawing candidate inferences about the overall relationship between what is depicted in the two diagrams. In contrast, our work constructs a structural model from a diagram by analogical transfer.

The origin of our SBF models lies in Chandrasekaran's Functional Representation (FR) scheme for representing the functioning of devices [22, 3]. The Torque program [13, 14] completed a partial SBF model of the stretching of a spring system by analogy to the SBF model of a flexible beam. However, Torque did not contain any spatial representation of either the spring system or the flexible beam. The Archytas program's DSSBF models explicitly relate SBF models of physical devices with their spatial representations.

Narayanan and Chandrasekaran [18] have investigated the role of spatial representations in behavioral simulation of physical devices, and proposed the use of images and symbolic structures as representations that may enable both spatial and propositional inferences. More recently, Chandrasekaran [2] has proposed multimodal internal representations as a central element of the cognitive architecture. Our DSSBF models might be viewed as an example of such multimodal representations.

5 Conclusions

A persistent and central question in diagrammatic reasoning is how an agent understands a diagram, where we take understanding of a diagram to be the construction of a model. This is a very complex, multifaceted question. We posit that for domains which are familiar to an agent, the agent might construct a model of a diagrammatic representation by analogical transfer of the model of a similar diagrammatic representation. For familiar situations, the agent already may have constructed models of diagrammatic representations it has encountered in the past, and encapsulated the diagram and the model as a case in its memory. For example, an expert engineer who specializes in the design of kinematic devices already might have constructed functional models of many known designs. Further, for familiar situations, the agent may find a source case in its memory that strongly resembles a target problem. For instance, given a target diagram of a kinematic device, a design specialist in kinematics design might find in its memory a source case that is very similar to the target diagram. When this happens, the agent may construct a teleological model of the target by analogical transfer of the model from the source case.

Translating the above high-level account of construction of a model from a diagram into an operational computer program raises a number of challenging process and content questions. We do not presently have a complete answer to all the hard process or content issues. Instead, in this paper, we have sketched outlines of (1) a content account of DSSBF models of physical devices that relate drawings to shape, shape to structure, structure to behavior, and behavior to function, and (2) a process account of analogical mapping between the shape-level representations of the target and the source drawings, in which knowledge about the function of shapes helps guide the mapping process. We have also shown that this content and process actually works for transferring a teleological model from a source drawing to a target drawing at least for a problem in which the distinctions between the source and the target diagrams make for no difference in the structural models of the source and the target.

Acknowledgments

This research has been supported in part by a NSF (IIS) grant (Award number 0534266) on Multimodal Case-Based Reasoning in Modeling and Design. This paper has significantly benefited from critiques by anonymous reviewers of earlier drafts.

References

- [1] C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.
- [2] B. Chandrasekaran. Multimodal perceptual representations and design problem. In J. S. Gero, editor, *Visual and Spatial Reasoning in Design*. Key Centre of Design Computing and Cognition, University of Sydney, NSW, Australia, 1999.
- [3] B. Chandrasekaran, A. K. Goel, and Y. Iwasaki. Functional representation as design rationale. In *Proc. European Workshop on Case-Based Reasoning*, pages 58–75, 1993.
- [4] C.-W. K. Chen and D. Y. Y. Yun. Unifying graph matching problems with a practical solution. In *Proc. Int'l Conf. on Systems, Signals, Control, and Computers*, 1998.
- [5] T. Dar, L. Joskowicz, and E. Rivlin. Understanding mechanical motion: From images to behaviors. *Artificial Intelligence*, 112:147–179, 1999.
- [6] T. G. Evans. A heuristic program to solve geometric analogy problems. In M. Minsky, editor, *Semantic Information Processing*. MIT Press, Cambridge, MA, 1968.
- [7] B. Falkenhainer, K. D. Forbus, and D. Gentner. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41:1–63, 1990.
- [8] R. W. Ferguson and K. D. Forbus. Telling juxtapositions: Using repetition and alignable difference in diagram understanding. In K. Holyoak, D. Gentner, and B. Kokinov, editors, *Advances in Analogy Research*, pages 109–117. New Bulgarian University, Sofia, Bulgaria, 1998.
- [9] R. W. Ferguson and K. D. Forbus. GeoRep: A flexible tool for spatial representation of line drawings. In *Proc. 17th National Conf. on Artificial Intelligence (AAAI-2000)*, Austin, Texas, 2000. AAAI Press.
- [10] A. K. Goel. A model-based approach to case adaptation. In K. J. Hammond and D. Gentner, editors, *Proc. 13th Annual Conf. of the Cognitive Science Society*, pages 143–148. Lawrence Erlbaum Associates, August 1991.
- [11] A. K. Goel. Adaptive modeling. In Y. Iwasaki and A. Farquhar, editors, *Qualitative Reasoning: Papers from the 10th Annual Workshop*, Technical Report WS-96-01, pages 67–73. AAAI Press, Menlo Park, California, 1996.
- [12] A. K. Goel and B. Chandrasekaran. Functional representation of designs and redesign problem solving. In *Proc. 11th International Joint Conf. on Artificial Intelligence (IJCAI-89)*, pages 1388–1394. Morgan Kaufmann, 1989.
- [13] T. Griffith, N. Nersessian, and A. K. Goel. The role of generic models in conceptual change. In *Proc. 18th Annual Conf. of the Cognitive Science Society*. Lawrence Erlbaum Associates, 1996.
- [14] T. Griffith, N. Nersessian, and A. K. Goel. Function-follows-form transformations in scientific problem solving. In *22nd Annual Conf. of the Cognitive Science Society*, pages 196–201. Lawrence Erlbaum Associates, 2000.
- [15] D. Hofstadter and G. McGraw. Letter spirit: Esthetic perception and creative play in the rich microcosm of the roman alphabet. In D. Hofstadter and the Fluid Analogies Research Group, editors, *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*, chapter 10, pages 407–466. Basic Books, 1995.

- [16] L. Joskowicz. Shape and function in mechanical devices. In *6th National Conf. on Artificial Intelligence (AAAI-87)*, pages 611–615. AAAI Press, 1987.
- [17] G. McGraw and D. Hofstadter. Perception and creation of diverse alphabetic styles. In S. Harrison, editor, *Artificial Intelligence and Creativity: Papers from the 1993 Spring Symposium*, Technical Report SS-93-01, pages 11–18. AAAI Press, Menlo Park, California, 1993.
- [18] N. H. Narayanan and B. Chandrasekaran. Reasoning visually about spatial interactions. In *Proc. 12th International Joint Conf. on Artificial Intelligence (IJCAI-91)*, pages 360–365. Morgan Kaufmann, 1991.
- [19] N. H. Narayanan, M. Suwa, and H. Motoda. How things appear to work: Predicting behaviors from device diagrams. In *Proc. 12th National Conf. on Artificial Intelligence (AAAI-94)*, pages 1161–1167. AAAI Press, 1994.
- [20] G. S. Novak. Diagrams for solving physical problems. In J. Glasgow, N. H. Narayanan, and B. Chandrasekaran, editors, *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, pages 753–774. AAAI Press/The MIT Press, Menlo Park, CA, 1995.
- [21] J. W. Raymond, E. J. Gardiner, and P. Willett. Heuristics for similarity searching of chemical graphs using a maximum common edge subgraph heuristic. *J. Chem. Inf. Comput. Sci.*, 42:305–316, 2002.
- [22] V. Sembugamoorthy and B. Chandrasekaran. Functional representation of devices and compilation of diagnostic problem-solving systems. In J. Kolodner and C. Riesbeck, editors, *Experience, Memory, and Reasoning*, pages 47–73. Lawrence Erlbaum, Hillsdale, NJ, 1986.
- [23] T. F. Stahovich, R. Davis, and H. Shrobe. Generating multiple new designs from a sketch. *Artificial Intelligence*, 104(1–2):211–264, 2001.