Using AI Planning Techniques to Automatically Generate Image Processing Procedures: A Preliminary Report

Steve Chien
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, M/S 525-3660
Pasadena, CA 91109-8099
chien@aig.jpl.nasa.gov

Abstract

This paper describes work on the Multimission VICAR Planner (MVP) system to automatically construct executable image processing procedures for custom image processing requests for the JPL Multimission Image Processing Lab (MIPL). This paper focuses on two issues. First, large search spaces caused by complex plans required the use of hand encoded control information. In order to address this in a manner similar to that used by human experts, MVP uses a decomposition-based planner to implement hierarchical/skeletal planning at the higher level and then uses a classical operator based planner to solve subproblems in contexts defined by the high-level decomposition. Second, the image processing domain is characterized by large amounts of search to find the correct program options for images (e.g. operator effects), rather than search among different programs (e.g. planning operators) and many of these program options are incompatible (i.e. certain combinations cannot be used). MVP represents these interactions by using codesignation constraints to specify program options for operators and allowing these constraints to occur in operator preconditions allowing MVP to search the program option space efficiently while handling negative interactions between program options.

1. Introduction

This paper describes a planning system being fielded to automatically generate image processing procedures to satisfy science requests for image data made to the Multimission Image Processing Laboratory (MIPL) run by the Jet Propulsion Laboratory. Currently, a group of human experts, called analysts, receive written requests for science data processed and formatted in a certain manner. These analysts then determine the relevant data and appropriate image processing steps required to produce the requested data and write an image processing program in a programming language called VICAR[1](LaVoie et al. 1989).

Unfortunately, this current mode of operations is extremely labor-intensive and knowledge intensive. This task is labor intensive in that constructing the image processing procedures is a complex, tedious process which can take anywhere from several hours to several months of effort. There are currently 3 groups of analysts, each of 10-20 analysts, whose primary task is to construct these VICAR programs. Many other users at JPL and other sites also write VICAR scripts, with the total user group numbering in the hundreds. The VICAR procedure generation problem is also a knowledge intensive task. In order to construct VICAR procedures an analyst must posses knowledge of:

1. image processing and image processing programs (as of 1/93 there were approximately 50 frequently used programs, some having as many as 100 options)
2. database organization and database label information to understand the state of relevant data
3. the VICAR programming language to produce and store relevant information.

Because of the significant amount of knowledge required to perform this task, it takes several years for an analyst to become expert in a VICAR image processing area.

In developing the MVP system, we encountered two major difficulties in applying conventional classical planning technology. First, the complexity and length of plans being generated required the use of encoded control knowledge to allow the planner to find plans within a reasonable time limit. Second, VICAR programs frequently have many program parameters, called options, which must be set correctly to specify the exact context in which the program is being used. Searching for the correct program option settings in an efficient manner while representing interactions between program options is key to efficient VICAR planning.

In order to encode control knowledge to guide the search for plans, MVP uses a combination of several planning paradigms. At a higher level, MVP accepts a set

---

This paper describes work performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. Other past and present members of the MVP team are Helen Mortensen, Christine Ying, Shouyi Hsiao, Alex Gray, Joe Nieten, and Jean Lorre.

[1] For Video Image Communication and Retrieval which actually is a misnomer as VICAR is used to process much non-video image data such as Magellan Synthetic Aperture Radar image data.

of image processing goals and uses skeletal and hierarchical planning techniques to classify it into one of a set of problem classes, such as movie-frame color triplet processing, or mosaicking with absolute navigation. This problem classification then allows the individual processing goals to be assigned into different subproblems based upon the overall context of the image processing task (i.e. problem class, project/spacecraft, presence or absence of other goals, etc.). This process ends when the planner is able (after search) to reduce all of the high level goals into goals achievable by the operator-based planner (Pemberthy & Weld 1992). Each of these subproblems is then solved by the operator-based planner. The resulting set of plan operators is then converted to an executable PDF by a code generation module which uses macro expansion to perform syntactic modifications to produce the desired output.

The VICAR domain also has the characteristic of search among program options (operator effects) to achieve goals. For example, one type of correction is a rotational perspective correction which corrects for the planetary rotation when combining several images taken at different times into a single image. In order to perform this correction, the VICAR program PTP must know the position of the camera (e.g., spacecraft) relative to the planet center. This information can be specified (derived) using any one of several sources: through navigation predict information (e.g. spacecraft navigation information), own created navigation information (e.g., information derived by analyzing the edge of the planet in the image), own supplied spacecraft pointing information (e.g. previously navigating and then processing the image). Additionally, each of these navigation methods has further choices and options, (e.g. the own created navigation information requires that you specify the navigation source used, which may be one of 8 source methods). The VICAR language uses program options to specify to the programs how this information is specified. Each set of program options typically corresponds to a set of preconditions which will allow the operator to achieve some effects. Unfortunately, these program options often have negative interactions - i.e., certain combinations of program options are incompatible. Because of the number and complexity of these program options and their interactions, frequently MVP is searching to find a consistent set of program options (whose preconditions can be satisfied) rather than to find an operator whose preconditions can be satisfied.

In order to deal with this search among program options MVP specifies program options as codesignation constraints on variables occurring in planning operators. These codesignations then appear in preconditions of effects and thusly positive and negative interactions between program options can be directly represented and reasoned about using least commitment strategies. This means that MVP need not commit to program option settings unnecessarily - when determining the correct setting for one option, it need not constrain other unrelated options. This also means that when enforcing protections, MVP can enforce that incompatible options not be used by enforcing negative codesignations to prevent preconditions of the possibly interfering effect.

The remainder of this paper is organized as follows. Section 2 describes the VICAR image processing domain in greater detail and describes the overall operations context and architecture of the MVP system. Section 3 describes how MVP integrates decomposition-based and operator-based planning and how this allows MVP to operate in a manner more understandable to the analysts. Section 4 describes how the representation of VICAR program options as codesignations and codesignations in preconditions allows MVP to efficiently reason about VICAR program options. Section 5 reports on the current status of the MVP system and describes some characteristics of the domain(s) implemented. Section 6 describes current and future work an summarizes the principle contributions of this paper.

2. VICAR Image Processing and MVP

VICAR is a general-purpose image processing programming language designed to promote the development and re-use of general-purpose image processing algorithms for MIPL needs. The primary function of VICAR is to allow individual image processing steps (called VICAR programs) to be combined into more complex image processing scripts called procedure definition files (PDFs). MIPL analysts construct PDFs to perform image correction, image enhancement, construct mosaics, and to create movies and render objects. Individual processing programs perform functions such as photometric correction (correcting the image for lighting conditions due to the position of the sun relative to he camera and target), radiometric correction (correcting for varying camera response depending on where in the field of view the image is read), and line fillin (replacing missing lines cause by data transmission errors by interpolation).

VICAR image processing maps naturally onto the AI planning problem where: 1) the initial state corresponds to the initial database state (the state of relevant image files, the existence of appropriate calibration files, etc.); 2) planning operators correspond to VICAR programs; and 3) the problem goals correspond to the image processing goals (desired image characteristics and format, etc.). The VICAR image processing domain represents a rich AI planning domain with tens of relevant database label fields relevant to processing, approximately 100 VICAR programs (as of 1/93), many of which have tens of program options which control the image processing effects of the program, with many subareas of VICAR image processing, with diverse sets of problem goals, (tens per subarea).

Due to the diversity of VICAR image processing, we are currently targeting the VOYAGER and GALILEO mosaicking and color triplet processing areas of image processing. This allows us to focus on a subset of the relevant database fields, VICAR programs and options, and problem goals. This particular target area tracks approximately 50 image file attributes, 30 VICAR programs, and 20 image processing goals. We estimate that there are on the order of tens of these VICAR processing subdomains and are currently evaluating several follow-on application areas.

The overall architecture for the MVP system is shown in Figure 1. The user inputs a problem specification consisting of processing goals and certain image information using a menu-based graphical user interface. These goals and problem context are then passed to the decomposition-based planner which uses skeletal and hierarchical planning methods to classify the problem type and use this classification to decompose the problem into smaller subproblems. During this decomposition process, MVP determines which information on the database state is needed by the planner to solve the subproblems.

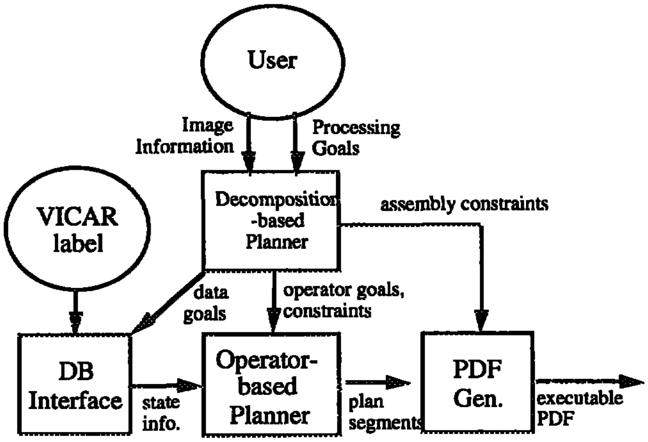

Figure 1: MVP Architecture

These subproblems are then solved by a conventional operator-based planner using the subproblem goals and initial states as indicated by the problem decomposition. The resulting plan segments are then assembled using constraints derived in the decomposition process. The resulting plan is then used to generate an actual executable VICAR PDF using conventional macro-expansion techniques.

3. Integrating Decomposition-based and Operator-based Planning

Plans in the MVP domain can be of considerable length (up to 100 steps) and each step (or VICAR program) can involve reasoning about numerous complex effects (many operators have tens of effects). Due to the large search space caused by this complexity, conventional operator-based planning approaches are not able to tractably construct plans in the VICAR domain without significant control knowledge.

Additionally, even if a purely operator-based planning approach were able to generate plans to solve the VICAR problems, these plans would be difficult for MIPL analysts to understand. Typically, analysts begin by classifying the general problem being addressed into one of a general class of problems, such as mosaicking, color triple processing, etc. They then use this classification and the problem context to decompose the plan into several abstract steps, such as local correction, navigation, registration, touch-ups, etc. A planning system which mimicked this approach to producing VICAR PDFs would be desirable.

Skeletal and Hierarchical Planning in MVP

Skeletal planning (Iwasaki & Friedland 1985) is an approach to planning which casts planning as a structured classification problem. In skeletal planning, a planner identifies a new problem as one of a general class of problems based upon the goals and initial state. This technique was originally developed as a model of experiment design in molecular biology; however, skeletal planning is also an accurate model of how expert analysts attack VICAR procedure generation problems. Typically, in a VICAR problem, there is a central goal for processing, such as mosaicking, which then dictates a decomposition of the overall problem into subproblems such as local correction, navigation, and registration. MVP attacks a VICAR problem by first determining the general problem class, and then using this problem class to perform an initial decomposition of the top--level image processing goals.

Hierarchical planning (Stefik 1981) is an approach to planning where abstract goals or procedures are incrementally refined into more and more specific goals or procedures as dictated by goal or procedure decompositions. MVP uses this approach of hierarchical decomposition to refine the initial skeletal plan into a more specific plan specialized based on the specific current goals and situation. This allows the overall problem decomposition to be influenced by factors such as the presence or absence of certain image calibration files or the type of instrument and spacecraft used to record the image. For example, geometric correction uses a model of the target object to correct for variable distance from the instrument to the target. For VOYAGER images, geometric correction is performed as part of the local correction process, as geometric distortion is significant enough to require immediate correction before other image processing steps can be performed. However, for GALILEO images, geometric correction is postponed until the registration step, where it can be performed more efficiently.

Decomposition-based Planning in MVP

MVP uses a decomposition-based approach (Lansky 1992) to perform Skeletal and Hierarchical planning. In a decomposition-based approach, decomposition rules dictate how in plan-space planning, one plan can be legally transformed into another plan. The planner then searches the space plans defined by these decompositions. Decomposition-based approaches are extremely powerful in that many other paradigms (such as modal truth criterion planning (Lansky 1992) can be implemented in a decomposition-based approach. Syntactically, a decomposition rule is of the form:

LHS		RHS
G_I = initial goal set/actions		G_R = reduced goal set/actions
C_0 = constraints	==>	C_I = constraints
C_2 = context		N = notes on decomposition

This rule states that a set of goals or actions G_I can be reduced to a new set of goals or actions G_R if the set of constraints C_0 is satisfied in the current plan and the context C_2 is satisfied in the current plan provided the additional constraints C_I are added to the plan.

Skeletal planning in MVP is implemented in by encoding decomposition rules which allow for classification and initial decomposition of a set of goals corresponding to a VICAR problem class. The LHS of a skeletal decomposition rule in MVP corresponds to a set of conditions specifying a problem class, and the RHS specifies an initial problem decomposition for that problem class. For example, the following rule represents a decomposition for the problem class mosaicking with absolute navigation.

LHS	RHS
G_I= mosaicking goal present	G_R = 1. local correction, 2. navigation 3. registration 4. mosaicking 5. touch-ups
C_0= null	C_I = these subtasks be performed in order 1. 2. 3. 4. 5. protect local correction until mosaicking
C_2= an initial classification has not yet been made	N = the problem class is mosaicking

This simplified decomposition rule states that if mosaicking is a goal of the problem and an initial problem decomposition has not yet been made, then the initial problem decomposition should be into the subproblems local correction, navigation, etc. and that these steps must be performed in a certain order. This decomposition also specifies that the local correction goals must be protected during the navigation and registration processes.

In general, MVP permits goals and abstract steps to be specified in the G_I & G_R fields. The constraints C_0 & C_I may be ordering and codesignation constraints and the context may specify the presence or absence of attributes over the plan or goals (such as a certain goal not being present, etc.).

Hierarchical planning is also implemented within the decomposition framework. In this case the LHS specifies a context in which a set of goals or actions can be decomposed into a lower level set of goals or actions. For example, the decomposition rule below states that if the limb is present in all of the images (meaning that the sun-facing edge of the planet is visible in all of the images), for VOYAGER and GALILEO images, the navigation step can be performed by absolute navigation (a process in which each of the images can be navigated independently).

LHS	RHS
G_I= navigation action present	G_R = 1. absolute navigation
C_0= null	C_I = null
C_2= the project is VOYAGER or GALILEO and limbs are present in all images	N = null

This decomposition-based approach to skeletal and hierarchical planning in MVP has several strengths. First, the decomposition rules very naturally represent the manner in which the analysts attack the procedure generation problem. Thus, it was a relatively straightforward process to get the analysts to articulate and accept classification and decomposition rules for the subareas which we have implemented thus far. Second, the notes from the decomposition rules used to decompose the problem can be used to annotate the resulting PDF to make the VICAR programs more understandable to the analysts. Third, relatively few problem decomposition rules are easily able to cover a wide range of problems and decompose them into much smaller subproblems.

In the current version of MVP, there are on the order of 10 skeletal decomposition rules and 30 hierarchical decomposition rules which cover on the order of hundreds of goal combinations and problem contexts. These decomposition rules are able to break down script into typically 5 goal sets each of approximately 5 to 10 goals, where each goal set is typically achievable by a subplan of 10 operators or less. This size of subplan is easily handled by the operator-based planner with search of on the order of thousands of plans and can be constructed on the order of 10s of seconds for a Sparcstation 10.

4. Program Options in VICAR and MVP

One interesting aspect of the VICAR domain is that the majority of the search to achieve goals and to enforce protections is not at the program selection level (which corresponds to operator selection in the planning process) but rather at the program option level (which corresponds to the operator effect planning level). Thus, when planning to achieve a goal, MVP searches more in determining how to set program options to achieve a goal (e.g. how to set variable constraints to satisfy preconditions) rather than in determining which VICAR program (planning operator) to use to achieve the goal. This presents a problem for efficiently reasoning about interacting program options (operator effects) in that certain combinations of program options (operator effects) are inconsistent (i.e., cannot be used together). Searching these combinations of operators effects efficiently when the operator effects do not interact, yet correctly restricting to those legal combinations is novel to the MVP planner.

Due to this difficulty of search among VICAR program options, MVP uses an operator-based planning component which extends conventional operator-based planning (Pemberthy & Weld 1992). MVP represents VICAR program options as variable codesignation constraints. Thus, if a VICAR program has a program option which allows for several ways to specify spacecraft pointing information for a particular image processing step, MVP would represent these different methods as conditional effects of a single planning operator, with the appropriate preconditions (including variable codesignations). If certain program options (operator effects) are inconsistent, they would be represented by having preconditions with conflicting codesignation constraints. When using an operator effect to achieve a subgoal in the plan, MVP first checks to see if the codesignation preconditions are consistent with the plan, only then allowing the effect to be used (and adding the codesignation constraints to the plan).

For example, one VICAR program is the PTP program, which allows for multiple images taken at similar times to be corrected to appear as if they were taken at the same time. This program needs to know the position of the spacecraft relative to the target of the image (typically the planet center). This information can be specified in one of several different ways, such as using the spacecraft navigation information, specific VICAR programs which attempt to compute this information from the image (the usual method), or by specifying the exact pixel location known from previous operations in the PDF. Typically, an analyst will include VICAR code to derive this information directly from the image. In this case the exact program and options being used to compute this information are frequently needed by the PTP program. For example, for the VOYAGER project, if one wishes to use pointing information previously derived using the FARENC program, it would be stored in a navigation data structure called SEDR. The conditional effect might look something like the following (codesignation constraints are marked by an asterisk *).

IF
(SEDRSRC is specified to be FARENC)* and
(PC and RPC are not specified)*
the project of *file* is VOYAGER1 or VOYAGER2 and
appropriate SEDR data files for *file* exist and
the camera number RCAM for the *file* has been correctly specified and
the FDS for *file* has been correctly specified
THEN
then output image *outfile* will be registered to the reference image as specified

This method for representing VICAR program options is important in that it allows for independent program options to be reasoned about and constrained independently yet represents the interaction between interacting options.

For example, the PTP program option to translate the image during the PTP step, requires that the camera pointing specification be directly specified using the planet center (PC) and reference planet center options (RPC), which specify a particular point in the image directly as the planet center. These options are incompatible with the FARENC source of camera pointing information. MVP represents this constraints by negative codesignations appearing in the preconditions of these incompatible options (the *-ed codesignation constraints listed above).

However, non-interacting options such as PTP options to resize the image or to include or delete the background of the image are not affected. These options do not interact with the specification of pointing information and thusly can be reasoned about independently.

In contrast, most planners do not allow for codesignation constraints on operator effects, and thusly would have to place contradictory preconditions to enforce disallowed combinations or break inconsistent operator effects into different planning operators - representing consistent combinations of operator effects. Detecting inconsistent preconditions when choosing an effect is analogous to our codesignation method. Not detecting these contradictory preconditions when choosing an effect would cause considerable unnecessary search. Breaking inconsistent effects (program options) into separate operators requires an increase in the number of operators exponential in the number of inconsistent options (N pairs of incompatible options requires 2^N operators). Even worse, when selecting an operator which one option decided, the planner would have to arbitrarily commit to decisions on other program options - potentially causing unnecessary search. As the number of program options can be quite large (frequently in the tens of options and sometimes as many as 100 options), these are important representational and search efficiency issues.

5. Status of MVP

MVP version 1.0, was demonstrated in June 1993 and addressed the subproblem of mosaicking with absolute navigation for VOYAGER images. This system handled approximately 10 goals, involving approximately 15 VICAR programs, and tracked about 20 file features. In December 1993, MVP version 1.1 was demonstrated and had additional capabilities allowing addressing GALILEO color triplet processing and mosaicking as well as some simple filtering steps. Version 1.1 handles approximately 20 goals, 30 VICAR programs, and models approximately 50 file attributes (Chien & Mortensen, 1993).

MVP version 1.0 and 1.1 were implemented in Lucid Common LISP, and run under the Openlook windowing environment. MVP version 2.0, which has roughly equivalent domain coverage to version 1.1 is implemented in C, and run in the Motif windowing environment (the MIPL standard) and will be operational in April 1994.

The current coverage for MVP is already at the useful level. Over a test suite of 5 typical mosaicking and color reconstruction tasks, an expert analyst estimated that MVP would reduce effort to generate an initial PDF for an expert analyst from 1/2 a day to 15 minutes and that it would reduce the effort for a novice analyst from several days to 1 hour. For one particularly challenging PDF, the expert analyst estimated the complete start-to-finish task took approximately 3 days effort, and that with MVP the task would have taken less than half that time.

6. Discussion and Conclusions

There are a number of outstanding research issues which prevent straightforward application of MVP to certain other VICAR tasks. In this section we discuss several of these areas of current and future work. One important issue is that of representing plan quality. Frequently in the VICAR domain, there are multiple ways of achieving processing goals, but the quality of the final resulting image will vary depending upon which approach is used. For example, when constructing a mosaic using relative navigation (a process of matching adjacent frames by finding common points), the order in which one places the tiles together will significantly affect final image quality. In other cases, performing the correction steps in different orders, although possible, will result in varying image quality. Currently, these preferences are represented by using decompositions and false preconditions, but more general declarative methods for representing this information would be preferable because of added ease of maintenance.

Another area of work is that of automatically recognizing and handling simple loops. Frequently a set of images are processed in an identical or almost identical fashion. The current MVP implementation is able to handle a number of these cases in the decomposition phase, however a more general approach would be to recognize the common goal structure and re-use the plan found for the first image re-instantiated for the other images. Currently MVP will generate plans which contain the loops expanded out. While recognizing the loops would improve the speed of plan generation, the more significant advantage is that MVP could produce more readable plans containing the loops. We believe that straightforward EBL techniques applied to constraint-posting planning techniques (Chien 1990) would allow for this capability.

An additional area for work is creating a development environment to allow for maintaining and extending domain knowledge expressed in MVP. This would involve tools for tracing and debugging similar to those currently available for rule-based systems. Such tools would allow analysts to easily assess the effects of modifying operator definitions to reflect program changes and to debug and refine operator definitions when encoding new application domains. Another useful capability would be explanation facilities for the planner. Frequently an analyst wants to know if another operator ordering is consistent with the current plan structure (dependencies) or why operator X was used in the plan (in particular, why wasn't a different operator Y used?). Developing an environment to support these capabilities (particularly to analyze other possible plan structures) is an area of future work.

One major area of current work is extending the current MVP system to other application areas. We are currently examining earth imaging applications and atmospheric science applications of VICAR image processing as potential future application areas.

In summary, this paper has described an application of classical planning techniques to automatic generation of image processing procedures. In using classical AI planning techniques for VICAR image processing, two difficulties were encountered. First, the length and complexity of VICAR procedures necessitated encoding of control knowledge to allow for tractable solution of real VICAR problems. In order to solve these problems in a manner understandable by human analysts, MVP first decomposes the problems, attacking them in a skeletal planning and hierarchical planning methodology. The resulting subproblems are then solved by a conventional operator-based planner. The second difficulty encountered is that in the VICAR application domain, frequently MVP must perform significant search among program options (in contrast to typical planning applications where the search is among planning operators). To address this problem, MVP represents VICAR program options as codesignation constraints on variables and allows codesignation constraints to appear in operator preconditions. As a result, MVP can reason efficiently about program options in a least commitment fashion. Finally, we described the current status of the MVP planner and outlines several areas of current and future work.

REFERENCES

(Chien 1990) S. Chien, "An Explanation-based Learning Approach to Incremental Planning," Technical Report UIUCDCS-R-90-1646, Department of Computer Science, University of Illinois, November 1990.

(Chien & Mortensen, 1993) S. Chien and H. Mortensen, "MVP Domain Specification Document, Version 1.0 - Mosaicking with Absolute Navigation and Color Triplet Processing," Internal JPL Publication, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 1989.

(Iwasaki & Friedland, 1985) Y. Iwasaki and P. Friedland, "The Concept and Implementation of Skeletal Plans," Automated Reasoning 1, 1 (1985), pp. 161-208.

(Lansky 1993) A. Lansky, "Localized Planning With Diversified Plan Construction Methods," Technical Report FIA-93-17, NASA Ames Research Center, June 1993.

(LaVoie et al. 1989) S. LaVoie, D. Alexander, C. Avis, H. Mortensen, C. Stanley, and L. Wainio, VICAR User's Guide, Version 2, JPL Publication D-4186, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 1989.

(Pemberthy & Weld, 1992) J. S. Pemberthy and D. S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," Proceedings of the Third International Conference on Knowledge Representation and Reasoning, October 1992.

(Stefik, 1981) M. Stefik, "Planning with Constraints (MOLGEN: Part 1)," Artificial Intelligence 16,2(1981), pp. 111-140.