MSL Mission Planning & Execution

Presenter: Tony Barrett

Designers/implementers:
Steve Chien
Dan Dvorak
Erann Gat
Kim Gostelow
Robert Keller
Russell Knight
Richard Morris
Robert Rasmussen
Thomas Starbird

1/15/04 Under the hood of MPE
Motivation

• A framework for planning, scheduling, and execution.
• A control centered approach toward representing plans as opposed to an action centered approach.
• This is the current default system for flying onboard the MSL rover.
Outline

- Plan/problem representation
  - Partially ordered constraints on state variables.
- MPE component architecture
  - Numerous threads of execution
- Elaboration & Scheduling
  - Subsumes both PO and HTN planning
- Plan execution
  - Firing time-points & enforcing constraints
The Problem

• Givens
  – A model of how state variables affect each other
  – Tactics for elaborating constraints into supporting constraints.
  – A temporally constrained set of commanded constraints on state variables.

• Objectives
  – Elaborate the constraints into an executable network, where constraints can be incrementally passed to controllers for each state variable.
  – Execute the elaborated network.
Plan/Problem Representation

- A network of timepoints connected by temporal and state-variable constraints.

This Goal will be in effect for at least 100 seconds, and no more than 900 seconds.

This Goal will be in effect for 600 seconds.
Plan/Problem Representation

• Definitions
  - Timepoint
    • A flexible point in time
  - Temporal Constraint
    • A timing relation between two Timepoints
  - Goal
    • A State Constraint over a time interval demarcated by two Timepoints
  - State Constraint
    • A representation of a set of values, used to specify a required (allowable) state
  - Executable goals (xgoal)
    • A merged state constraint that can be passed to a state variable controller

1/15/04

Under the hood of MPE
MPE Component Architecture

Constraint Network

Elaborators

State Variables

MPE Scheduler

Sensors

Estimators

Controllers

Actuators

1/15/04

Under the hood of MPE
Actual MPE Components
(each has an execution thread)

Under the hood of MPE
Elaboration

- Objective
  ➢ To set up conditions where a state variable's controller can enforce a goal's constraint.

- Algorithm
  Copy task network to proposed network for modification
  While there are goals to elaborate do
    Choose a goal $G$ to elaborate (from a heuristic ordering)
    Exhaustively choose elaboration tactic $E$ for $G$
    Apply $E$, which possibly generates new goals to elaborate
      - Backtrack if application of $E$ illegal

1/15/04 Under the hood of MPE
Defining Elaborators

- Default elaborators with GEL

(defGoal MyGoal (args)
  (between begin end)

(tactic MyFirstTactic

  (goal MySubGoal1 (args)
   (between nameTP1 beginTP) )
  (timeConstraint nameTP1 beginTP 10 20)

  (goal MySubGoal2 (args)
   (between beginTP endTP) )
  (timeConstraint beginTP endTP 5 15)
)

) // MyFirstTactic
) // MyGoal

(def tpl (mkTimepoint tpl))
(def tp2 (mkTimepoint tp2))
(elaborate (mkGoal MyGoal (args)
  (between tpl tp2))))

- Custom elaborators with C++

class MyFirstTactic : public Tactic
{
  ElaborationSpec* expand(TimePoint* beginTP, TimePoint* endTP)
  {
    addGoal(new MySubGoal1(),
     "nameTP1", beginTP);
    addTemporalConstraint(10, 20, "nameTP1", beginTP);

    addGoal(new MySubGoal2(),
     beginTP, endTP);
    addTemporalConstraint(5, 15,
     beginTP, endTP);
  }
};

class MyGoal : public Goal<BasicElaborator>
{
  Goal()
  {
    addTactic(new MyFirstTactic() );
    addTactic(new MySecondTactic() );
  }
};

1/15/04 Under the hood of MPE
Backtracking

- When an elaborator fails to elaborate
  - Scheduling/propagation fails
  - Cannot find a Tactic that works

- Reports failure to its parent

- Parent can tell a sibling of the failed Elaborator to re-elaborate
  - Doing things a different way could clear up conflicts with the sibling’s elaboration

- Then tell the failed Elaborator to “try again”

- Can do this for all SubGoals
Backtracking

Elaboration:

Communicating

Elaborator

Maintain attitude

Elaborator

Turn

Elaborator

elaborate

elaborate

elaborate

elaborate

Thrust

Elaborator

elaborate

elaborate

elaborate

Maintain Temp

10°-20° C

Elaborator

elaborate

elaborate

elaborate

Transition Temp

10°-20° C

Elaborator

Under the hood of MPE

1/15/04
Goal Net Scheduling

• Objective

➢ To merge goals into executable x-goals

• Algorithm (used during promotion)

For each state variable \textbf{SVAR} (from a heuristic ordering) do
  For each new goal \textbf{G} on \textbf{SVAR} (from a heuristic ordering) do
    Exhaustively choose \textbf{G}'s constrained start timepoint \textbf{S}
    Exhaustively choose \textbf{G}'s constrained end timepoint \textbf{E}
    Merge \textbf{G}[@\textbf{S,E}] into \textbf{SVAR}'s timeline – Backtrack if merge illegal
    Propagate the expected behavior of \textbf{SVAR} given the goals
    – Backtrack if the propagation illegal

Replace executing task network with proposed one if it scheduled

1/15/04 Under the hood of MPE
Computing X-Goals

Module Temperature:
- Transition to temp between X and Y degrees C
- Module Temperature: between X and Y degrees C
- Heater Switch & Health: healthy and switching between on and off
- Heater Switch & Health: healthy and switching between on and off

Under the hood of MPE
Computing X-Goals

Module Temperature: between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

Module Temperature: transition to temp between X and Y degrees C

Unconstrained
Computing X-Goals

宗态变量

Module Temperature: transition to temp between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

Module Temperature: between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

Unconstrained

1/15/04

Under the hood of MPE
Computing X-Goals

Module Temperature: transition to temp between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

Propagation tests

Module Temperature: transition to temp between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

1/15/04

Under the hood of MPE
Computing X-Goals

Module Temperature: between X and Y degrees C

Module Temperature: transition to temp between X and Y degrees C

Heater Switch & Health: healthy and switching between on and off

Heater Switch & Health: healthy and switching between on and off

Module Temperature: transition to temp between X and Y degrees C

Unconstrained

Under the hood of MPE
Computing X-Goals

Module Temperature:
transition to temp between X and Y degrees C

Heater Switch & Health:
healthy and switching between on and off

Module Temperature:
between X and Y degrees C

Heater Switch & Health:
healthy and switching between on and off

Under the hood of MPE
Computing X-Goals

Module Temperature: between $X$ and $Y$ degrees C

Heater Switch & Health: healthy and switching between on and off

1/15/04 Under the hood of MPE
Goal Net Execution

• Objective

➢ To feed x-goal constraints to state variable controllers as temporal constraints require and circumstances permit.

• Algorithm

For each unfired timepoint TP not temporally constrained into the future do
If TP's x-goals are ready to start or TP is about to time out then (fire)
For each x-goal G@[TP,*] respectively do
Send "start[G]" to G's state variable's controller & estimator
Inform G's parent goals to start or stop monitoring G's status
Timepoint A Fires

When A fires, estimator and controller get new instructions (a state constraint with timing information).

Temperature State Variable

State Constraint and Timing

Temperature Estimator

State Constraint and Timing

Temperature Controller

State Notification

XGoalsChecker
100 Seconds Later, Start Conditions are Monitored

Time: 3700

1/15/04

Under the hood of MPE
Timepoint B Fires

Temperature Estimator

State Constraint and Timing

Temperature State Variable

State Constraint and Timing

Temperature Controller

StateNotification

XGoalsChecker

New state constraint replaces previous one in controller and estimator

B will fire when temperature gets in range

Under the hood of MPE
Putting It All Together:
a high-level characterization of current MPE algorithm

• Elaboration algorithm
  Copy task network to proposed network for modification
  While there are goals to elaborate do
  Choose a goal G to elaborate (from a heuristic/random ordering)
  Exhaustively choose elaboration tactic E for G
  Apply E, which possibly generates new goals to elaborate
    – Backtrack if application of E illegal

• Scheduling algorithm (used during promotion)
  For each state variable SVAR (from a heuristic/random ordering) do
    For each goal G on SVAR (from a heuristic/random ordering) do
      Exhaustively choose G's constrained start timepoint S
      Exhaustively choose G's constrained end timepoint E
      Merge G@[S,E] into SVAR's timeline – Backtrack if merge illegal
      Propagate the expected behavior of SVAR given the goals
        – Backtrack if propagation illegal
  Replace executing task network with proposed one if it scheduled

• Execution algorithm
  For each unfired timepoint TP not temporally constrained into the future do
    If TP's Xgoals are ready to start or TP is about to time out then (fire)
      For each Xgoal G@[TP,+] respectively do
        Send "start[G]" to G's state variable's controller & estimator
        Inform G's parent goals to start or stop monitoring G's status

1/15/04
Under the hood of MPE
Next Steps in the Core

- Prioritized and delayed scheduling
  - Currently elaboration/scheduling just succeeds or fails.
  - Plan to break up commanded task subnet into a set of smaller prioritized subnets to elaborate and schedule in priority order. Subnets that fail to schedule persist for subsequent elaboration/scheduling attempts.

- State affects reasoning
  - Currently the side effects of commanded/elaborated constraints are not modeled.
  - Plan to embed a state-effects model into the system and use it to account for side effects during scheduling.
Observations

- The focus is on controlling to enforce constraints on state variables.
  - As opposed to the operator focus in AI planning.
  - Facilitates a clear way to merge concurrent action.

- The use of components and elaboration tasks enables a surprising amount of flexibility.
  - GEL can be used, but other approaches to elaboration are possible (in the same implementation).
  - The scheduler component can easily be replaced.

- Lifted time is the default, but it is not necessary.
References

- See:

http://mds.jpl.nasa.gov/outreach/