

Massachusetts Institute of Technology

Robust Execution of Contingent, Temporally Flexible Plans

Stephen Block
Andreas Wehowsky, Brian Williams

MIT Model-based Embedded & Robotic Systems CSAIL

Real World Autonomous Agents

- Coordinate multiple agents
- Provide robustness

MIT

Robustness to Disturbances

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution
- Communication latency : Distributed architecture
- Plan failure : Contingent mission plan

MIT

Robustness to Temporal Uncertainty

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan

Temporally flexible plans allow activities with *uncertain duration*

"Drive to rock"

Simple temporal constraint : $l \leq t^* - t \leq u$

$T = t^*$ $T = t^*$

S Drive to Rock E

Hardware command

MIT

Robustness to Execution Uncertainty

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution

Problem : Plan is either ...

- Brittle to temporal execution uncertainty
- Overly conservative to ensure success

Planning time

Temporally flexible plan

Assignment of execution times

Inflexible plan

Execution time

Plan execution

Hardware Commands

Hardware

MIT

Robustness to Execution Uncertainty

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution

Problem : Plan is either ...

- Brittle to temporal execution uncertainty
- Overly conservative to ensure success

Planning time

Temporally flexible plan

Maintenance of temporal flexibility

Execution time

Reactively schedule execution times

Hardware Commands

Hardware

MIT

Robustness to Execution Uncertainty

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution

Least commitment planning allows the executive to use temporal flexibility to respond to uncertainties at run time

Robustness to Execution Uncertainty

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution

Problem : Plan is either ...

- Brittle to temporal execution uncertainty
- Overly conservative to ensure success

Solution : Dispatchable execution ...

- Postpone scheduling until execution time

Requires two-stage execution ...

- Plan reformulation to compile the plan to a form for easy dispatching
- Dispatching to schedule and execute activities

Robustness to Communication Latency

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution
- Communication latency : Distributed architecture

Problem : Centralized architecture introduces communication bottleneck at master agent

Robustness to Communication Latency

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution
- Communication latency : Distributed architecture

Solution : A distributed architecture evens out the communication requirements

Distributed Architecture

Distributed Architecture

Contingent temporally flexible plan

Plan distribution

Reduced computational complexity

Avoids communication bottleneck

Plan Distribution

The diagram illustrates the process of plan distribution in a distributed system. On the left, five nodes (A1, A2, A3, A4, A5) are shown in a loose circle. A red arrow labeled "leader election" points from this group to a tree structure on the right. The tree structure shows A1 as the root, with A2 and A3 as children. A2 has children A4 and A5. Node A2 is highlighted with a red grid pattern, indicating it is the leader. Red arrows show the flow of information from A1 to A2 and A3, and from A2 to A4 and A5.

13

Plan Distribution

Exploit common hierarchical structure

if network is a single activity
 processor takes activity
 else if processor has followers
 distribute subnetworks to followers and self
 else if processor has co-leaders
 distribute subnetworks to coleaders and self
 else
 processor takes entire network

Exploit common hierarchical structure

```

if network is a single activity
    processor takes activity
else if processor has followers
    distribute subnetworks to followers and self
else if processor has co-leaders
    distribute subnetworks to coleaders and self
else
    processor takes entire network

```

The diagram is a directed graph with nodes and edges. Nodes are represented by colored circles: yellow (E), green (G), blue (F), blue (S), and red (T). Edges are labeled with mathematical expressions. The graph shows a complex network of dependencies between these components.

Robustness to Plan Failure

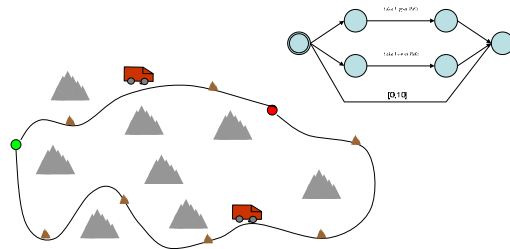
- Robustness to ...
 - Temporal uncertainty : Temporally flexible mission plan
 - Execution uncertainty : Dispatchable execution
 - Communication latency : Distributed architecture
 - Plan failure : **Contingent mission plan**

The diagram illustrates a contingent mission plan. It shows a path with a green start, a red stop, and a blue end, with a truck icon. The path is surrounded by grey mountains and brown triangles. A callout box shows a state transition diagram with nodes and edges labeled "Task plan (M)", "Task plan (B)", and "[M, B]".

15

Robustness to ...

- Temporal uncertainty : Temporally flexible mission plan
- Execution uncertainty : Dispatchable execution
- Communication latency : Distributed architecture
- Plan failure : **Contingent mission plan**



15

[illegible]

- Contingent temporally

Plan distribution

Reduced computational complexity

Plan selection

Planning time

Temporally flexible plan

Reduced computational complexity

Reformulation

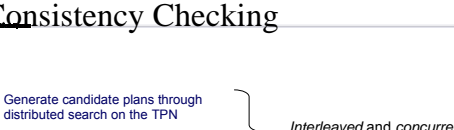
Avoids communication bottleneck

Dispatchable

Hardware Commands	
-------------------	--

Hardware

Interleaved Candidate Generation and Consistency Checking



1. Generate candidate plans through distributed search on the TPN
2. Test the generated plans for temporal consistency

Interleaved and concurrent

Implemented using a message passing scheme ...

findfirst Initial search for a consistent set of choice variable assignments
findnext Search for a new consistent assignment, to achieve global consistency
fail No consistent set of choice variable assignments was found
ack A consistent set of choice variable assignments was found

1. Generate candidate plans through distributed search on the TPN
2. Test the generated plans for temporal consistency

Interleaved and concurrent

Implemented using a message passing scheme ...

findfirst Initial search for a consistent set of choice variable assignments

findnext Search for a new consistent assignment, to achieve global consistency

fail No consistent set of choice variable assignments was found

ack A consistent set of choice variable assignments was found

13

Candidate Generation

1. Generate candidate plans through distributed search on the TPN
2. Test the generated plans for temporal consistency

Interleaved and concurrent

Implemented using a message passing scheme ...

findfirst Initial search for a consistent set of choice variable assignments
findnext Search for a new consistent assignment, to achieve global consistency
fail No consistent set of choice variable assignments was found
ack A consistent set of choice variable assignments was found

1. Generate candidate plans through distributed search on the TPN
2. Test the generated plans for temporal consistency

Interleaved and concurrent

Implemented using a message passing scheme ...

findfirst Initial search for a consistent set of choice variable assignments

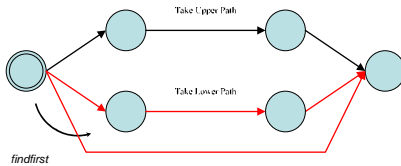
findnext Search for a new consistent assignment, to achieve global consistency

fail No consistent set of choice variable assignments was found

ack A consistent set of choice variable assignments was found

Candidate Generation

Nodes send *findfirst* messages to their children
Search progresses at increasing depth



19

Consistency Checking

1. Generate candidate plans through distributed search on the TPN
2. Test the generated plans for temporal consistency

Interleaved and concurrent

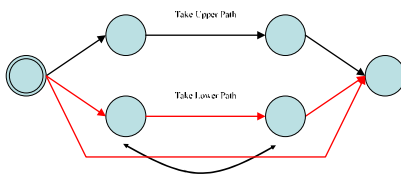
Implemented with synchronized distributed Bellman Ford Single Source Shortest Path algorithm

- Requires only local knowledge
- Uses a message passing scheme
- Runs on distance graph corresponding to active portions of the TPN
- Processor synchronization at every update cycle gives linear run time

20

Consistency Checking

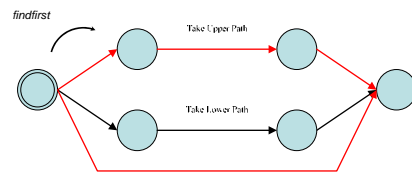
Nodes perform consistency checking and report to their parent
Checking progresses at decreasing depth



21

Candidate Generation

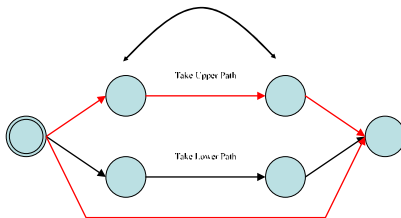
Nodes send *findfirst* messages to their children
Search progresses at increasing depth



22

Consistency Checking

Nodes perform consistency checking and report to their parent
Checking progresses at decreasing depth



23

Time Complexity Analysis

	Centralized Planner	DTP
Candidate Generation	N^e	N^e
Temporal Consistency Checking	$N^2 \log N + NM$	N
Overall Time Complexity	Exponential	Exponential

N Number of nodes
 M Number of edges
 e Size of domain of choice variables

- Worst case complexity of candidate generation corresponds to a plan entirely composed of choice nodes
- DTP uses parallel processing for parallel and sequence networks
- Time complexity much lower in realistic examples
- In practice, complexity of DTP is near-linear

24

Questions ?

