Intention Interleaving Via Classical Replanning

Mengwei Xu, Kim Bauters, Kevin McAreeavey, Weiru Liu
Extending Belief-Desire-Intention (BDI) Agents to Managing Intention Interleaving

- **Intention Resolution**: to avoid negative interference
  - Guarantee the achievability of intentions when interleaving the steps in different intentions

- **Intention Merging**: to facilitate positive interference
  - Perform one task once for at least two goals, i.e. “kill two birds with one stone”
Intention Resolution

Careless interleaving could result in that neither of its intention can be completed.
Motivation to Manage Intention Interleaving

Intention Merging

\[ G_1 \]
TransmitSoilResults

\[ a_1 \] EstablishConnection

\[ a_2 \] SendSoilResults

\[ a_4 \] BreakConnection

execute them once for both intentions

\[ a_1 \] EstablishConnection

\[ a_3 \] SendImageResults

\[ a_4 \] BreakConnection

\[ G_2 \]
TransmitImageResults
Belief-Desire-Intention: Literature

**Logics**
- [Cohen & Levesque, 1990]
- [Rao & Georgeff, 1991]
- [Shoham, 2009]

**Programming Languages**
- AgentSpeak [Rao, 1996]
- CAN [Winikoff et al., 2002]
- CANPLAN [Sardina et al., 2011]

**Software Platforms**
- Jason [Bordini et al., 2007]
- Jack [Winikoff, 2005]
- Jadex [Pokahr et al., 2013]
BDI Agent \( (\mathcal{B}, \Lambda, \Pi) \)

- **Initial belief base**
  Belief base specifying agent’s initial beliefs

- **Action library**
  Set of STRIPS-style action descriptions

- **Plan library**
  Set of plan rules
BDI Agent \((\mathcal{B}, \Lambda, \Pi)\)

**Initial belief base**
Belief base specifying agent’s initial beliefs

**Belief base** \(\mathcal{B} \subseteq \mathcal{L}\)
Set of formulas from logical language \(\mathcal{L}\)

\(\mathcal{B}\) must support:
- \(\mathcal{B} \models \varphi\) (Entailment)
- \(\mathcal{B} \cup \{\varphi\}\) (Addition)
- \(\mathcal{B} \setminus \{\varphi\}\) (Deletion)

Assume \(\mathcal{B}\) is a set of atoms
CAN: Agent $(\mathcal{B}, \Lambda, \Pi)$

**Action library**
Set of STRIPS-style action descriptions

**Action description** $\text{act} : \varphi \leftarrow \mathcal{B}^{-} ; \mathcal{B}^{+}$

- **Primitive action symbol**
- **Precondition** $\varphi \in \mathcal{L}$
- **Set of “add” atoms** $\mathcal{B}^{+} \subseteq \mathcal{L}$
- **Set of “delete” atoms** $\mathcal{B}^{-} \subseteq \mathcal{L}$
BDI Agent \((\mathcal{B}, \Lambda, \Pi)\)

Plan library
Set of plan rules

**Head**\((P)\): \(G\)
e.g. new goal

**Context**\((P)\): \(\varphi \in \mathcal{L}\)
Formula from \(\mathcal{L}\)

**body**\((P)\): \(h_1; \cdots; h_n\)
e.g. a sequence of actions or goals

Plan rule \(P = G: \varphi \leftarrow h_1; \cdots; h_n\)
BDI Operational Mechanism Sketch

Goal $G \xrightarrow{select} \text{Relevant Plans}$

$G : \varphi_1 \leftarrow P_1$

$G : \varphi_2 \leftarrow P_2$

$G : \varphi_3 \leftarrow P_3$

\vdots

$G : \varphi_n \leftarrow P_n$

$\subseteq \Pi$

\text{repeat for the subgoals}

$G : \varphi_{11} \leftarrow P_{11}$

$G : \varphi_{21} \leftarrow P_{21}$

$G : \varphi_{31} \leftarrow P_{31}$

\vdots

$G : \varphi_{n1} \leftarrow P_{n1}$

where $\mathcal{B} \models \varphi_{j1}, j \in \{1, \ldots, n\}$

A tree structure representing all possible ways of achieving a goal $G$
Our Intention Interleaving Framework in BDI

1. Intention Formalisation
   • Model an intention as an AND/OR graph
   • Define the execution trace for multiple intentions
   • Define the conflict-free and maximal-merged execution trace for multiple intentions

2. Intention Interleaving Planning Preparation
   • Indexing nodes
   • Defined terminal, initial node sets, and progression links of intentions
   • Computing overlapping programs between multiple intentions

3. Intention Interleaving Planning Formalism
   • Formalise FPP problem of interleaving intentions
   • Correctness Proof

4. Implementation

5. Evaluation
AND/OR Graphs for Intentions

\[ P_1 = G_1 : \varphi_1 \leftarrow a_1; a_2; a_4 \]

\[ P_2 = G_2 : \varphi_2 \leftarrow a_1; a_3; a_4 \]
Execution Trace for An Intention

To identifies every unique way in which a given intention can be achieved

Execution trace for $T_1$: $\tau(T_1) = G_1; P_1; a_1 ; a_2 ; a_4$

Execution trace for $T_3$: $\tau_1(T_3) = G_3; P_3; a_4; a_5$
$\tau_2(T_3) = G_3; P_4; b_4; b_5; b_6$
Execution Trace for Multiple Intentions

The construction of an execution trace of a set of intentions is to interleave elements in the execution traces of different intentions.

Potential execution trace for $T_1$ and $T_3$: $\sigma = G_1; P_1; G_3; P_3; a_1; a_4; a_2; a_4; a_5$

by interleaving $\tau(T_1) = G_1; P_1; a_1; a_2; a_4$ and $\tau_1(T_3) = G_3; P_3; a_4; a_5$
Execution Trace for Intentions (Cont.)

Conflict-free Execution Trace:

To model the successful interleaving which achieves all intentions

\[ \sigma[1] \uparrow \sigma[2] \uparrow \ldots \uparrow \sigma[j - 1] \uparrow \sigma[j] \uparrow \sigma[j + 1] \uparrow \ldots \uparrow \sigma[n] \]

where \( B_j \) is the belief base before the execution of the \( j^{th} \) element of an execution trace (i.e. \( \sigma[j] \))

An execution trace \( \sigma \) is conflict-free if and only if the following hold:

1. if \( \sigma[j] = P \in \Pi \), then \( B_j \models context(P) \), i.e. the context of plan \( P \) must be met before selection
2. if \( \sigma[j] = a \in \Lambda \), then \( B_j \models \psi(a) \), i.e. the pre-condition of action `a` must be met before selection
Mergeable Execution Trace of \( \{T_1, \ldots, T_m\} \)

To capture the overlapping programs of different intentions

\[
\sigma: \quad \sigma[1] \quad \sigma[2] \quad \cdots \quad \sigma[j] \quad \sigma[j + 1] \quad \cdots \quad \sigma[j + k - 1] \quad \sigma[j + k] \quad \cdots \quad \sigma[n]
\]

\( k \) consecutive same element from all different intentions in \( \sigma \)

\[
\sigma^m: \quad \sigma[1] \quad \sigma[2] \quad \cdots \quad \sigma[j] \quad \cdots \quad \sigma[j + k + 1] \quad \cdots \quad \sigma[n]
\]

An execution trace \( \sigma \) is a **mergeable execution trace** if and only if the following hold:

1. \( \exists j \in \{1, \ldots, n\} \) such that \( \sigma[j] = \sigma[j + 1] = \cdots \sigma[j + k] \) where \( 2 \leq k \leq n - j \);
2. \( \forall l \in \{1, \ldots, m\}, \forall s, t \in \{j, \ldots, j + k\} \) where \( s \neq t \) such that \( \sigma[s] \subseteq \tau(T_l) \subseteq \sigma \) and \( \sigma[t] \subseteq \tau(T_l) \subseteq \sigma \);
3. \( \sigma^m \) is a conflict-free execution trace where \( \sigma^m \) is the merged execution trace of \( \sigma \) by reducing each subsequence consisting of consecutive identical elements characterized by 1 and 2 in \( \sigma \) to only one element left.
Execution Trace for Intentions (Cont.)

Maximal-merged Trace of \( \{T_1, \cdots, T_m\} \)

To capture the most merged execution trace of multiple intentions

The merged execution trace \( \sigma^m \) of a mergeable execution trace \( \sigma \) of \( \{T_1, \cdots, T_m\} \) is maximal-merged if there is no another mergeable execution trace \( \sigma' \) of \( \{T_1, \cdots, T_m\} \) such that \( |\sigma'| < |\sigma^m| \) where \( |\sigma| \) stands for the length of \( \sigma \).

Perform action \( a_1 \) and \( a_4 \) once for both two goals \( T_1 \) and \( T_2 \).
Indexing Nodes

To ensure that e.g. the same actions in distinct plans is seen as different

A node $n$ is a top-level goal of intention $T$: $T(n)$

The nodes of actions and subgoals of intention $T$: $n^{P,j,T}$ to denote the $j^{th}$ member of $body(P)$ in $T$

A plan node in intention $T$: $n^T$

Initial node set for intentions $\{T_1, \cdots, T_m\}$: $z_0 = \{T_1(n), \cdots, T_m(n)\}$

Terminal node set for a goal node: a collection of the last element of each execution trace of a goal

Terminal node set for intentions $I = \{T_1, \cdots, T_m\}$

$z_g = \{tn_1, \cdots, tn_m\}$ where $tn_i$ is a terminal node of $T_i(n)$

$z_g \triangleright_{tn} I$ if $z_g$ is a terminal node set of $I$
Progression Links

To visualise the progression order of execution elements in the context of indexes

\[ \tau(T_1) : \begin{pmatrix} \text{node} & G_1 & P_1 & a_1 & a_2 & a_4 \\ \text{index} & T_1(\bar{n}) & P_{T_1}^1 & a_1^{P_{T_1}^1,1,T_1} & a_2^{P_{T_1}^2,2,T_1} & a_4^{P_{T_1}^3,3,T_1} \end{pmatrix} \]

The progression links of execution trace \( \tau(T_1) \)

\[
(T_1(\bar{n}) \rightarrow P_1^{T_1}) \\
(P_1^{T_1} \rightarrow a_1^{P_{1,1,T_1}}) \\
(a_1^{P_{1,1,T_1}} \rightarrow a_2^{P_{1,2,T_1}}) \\
(a_2^{P_{1,2,T_1}} \rightarrow a_4^{P_{1,3,T_1}}) 
\]

They are also called primitive progression links

\[ \tau(T_2) : \begin{pmatrix} \text{node} & G_2 & P_2 & a_1 & a_3 & a_4 \\ \text{index} & T_2(\bar{n}) & P_{T_2}^1 & a_1^{P_{T_2}^1,1,T_2} & a_3^{P_{T_2}^2,2,T_2} & a_4^{P_{T_2}^3,3,T_2} \end{pmatrix} \]

The progression links of execution trace \( \tau(T_2) \)

\[
(T_2(\bar{n}) \rightarrow P_2^{T_2}) \\
(P_2^{T_2} \rightarrow a_1^{P_{2,1,T_2}}) \\
(a_1^{P_{2,1,T_2}} \rightarrow a_3^{P_{2,2,T_2}}) \\
(a_3^{P_{2,2,T_2}} \rightarrow a_4^{P_{2,3,T_2}}) 
\]
Overlap Set of Multiple Intentions

To compute all potential overlapping programs among a set of intentions

The overlap set of \( \{T_1, \cdots, T_m\} \) is a set of tuples of the form \( \{(idx^1_b \rightarrow idx^1_e), \cdots, (idx^k_b \rightarrow idx^k_e)\} \) if:

1. \( J(idx^1_e) = \cdots = J(idx^k_e) \) where \( J(idx^i_e) \) represents the actual node of the ending index \( idx^i_e \);
2. \( \forall l \in \{1, \cdots, m\}, \exists s, t \in \{j, \cdots, j + k\} \) where \( s \neq t \) s.t. \( (idx^s_b \rightarrow idx^s_e) \in \tau(T_l) \) and \( (idx^t_b \rightarrow idx^t_e) \in \tau(T_l) \);

The progression links of execution trace \( \tau(T_1) \)

\[
\begin{align*}
(T_1(\bar{n}) \rightarrow P_1^{T_1}) & \quad (P_1^{T_1} \rightarrow a_1^{P_1,1,T_1}) & \quad (T_2(\bar{n}) \rightarrow P_2^{T_2}) & \quad (P_2^{T_2} \rightarrow a_1^{P_2,1,T_2}) \\
(a_1^{P_1,1,T_1} \rightarrow a_2^{P_1,2,T_1}) & \quad (a_2^{P_1,2,T_1} \rightarrow a_4^{P_1,3,T_1}) & \quad (a_1^{P_2,1,T_2} \rightarrow a_3^{P_2,2,T_2}) & \quad (a_3^{P_2,2,T_2} \rightarrow a_4^{P_2,3,T_2})
\end{align*}
\]

The overlap set of intention \( \{T_1, T_2\} \) has two elements as follows:

1. \( \{(P_1^{T_1} \rightarrow a_1^{P_1,1,T_1}), (P_2^{T_2} \rightarrow a_1^{P_2,1,T_2})\} \) where \( J(a_1^{P_1,1,T_1}) = J(a_1^{P_2,1,T_2}) = a_1 \);
2. \( \{(a_2^{P_1,2,T_1} \rightarrow a_4^{P_1,3,T_1}), (a_3^{P_2,2,T_2} \rightarrow a_4^{P_2,3,T_2})\} \) where \( J(a_4^{P_1,3,T_1}) = J(a_4^{P_2,3,T_2}) = a_4 \)
Overlap Progression Links

Let an element of overlap set of \( T \), \( \ldots \), \( T \) be \( ( idx^1_b \rightarrow idx^1_e ), \ldots , ( idx^k_b \rightarrow idx^k_e ) \).

Then we have a corresponding overlap progression link \( \alpha^O = ( \{ idx^1_b , \ldots , idx^k_b \} \rightarrow \{ idx^1_e , \ldots , idx^k_e \} ) \) such that the side of \( \alpha^O \) is \( size(\alpha^O) = k - 1 \), i.e. merging \( k - 1 \) extra primitive progression links.

by fault, the size of a primitive progression link \( \alpha^P \) is \( size(\alpha^O) = 0 \), i.e. no merging at all.

The overlap set of intention \( \{ T_1 , T_2 \} \) has two elements as follows:

1. \( \langle ( P_1^{T_1} \rightarrow a_1^{P_1,1,T_1} ) , ( P_2^{T_2} \rightarrow a_1^{P_2,1,T_2} ) \rangle \) \( \rightarrow \) \( \langle P_1^{T_1} , P_2^{T_2} \rangle \rightarrow \{ a_1^{P_1,1,T_1} , a_1^{P_2,1,T_2} \} \)

2. \( \langle ( a_2^{P_1,2,T_1} \rightarrow a_4^{P_1,3,T_1} ) , ( a_3^{P_2,2,T_2} \rightarrow a_4^{P_2,3,T_2} ) \rangle \) \( \rightarrow \) \( \langle a_2^{P_1,2,T_1} , a_3^{P_2,2,T_2} \rangle \rightarrow \{ a_4^{P_1,3,T_1} , a_4^{P_2,3,T_2} \} \)
A First-principles Planning (FPP) problem of interleaving intentions $I = \{T_1, \cdots, T_m\}$ is a tuple

$$\Omega = \langle \Sigma, X, O, s_0, S_G \rangle$$

- $\Sigma$ is a finite set of (propositional) atoms
- $X = \bigcup_{j=1}^{m} T_j(N_v \cup N_o)$ is the set of node indexes of $I$
- $O = O^p \cup O^o$ is a set of progression links
  - $O^p$ (resp. $O^o$) is the collection of primitive (resp. overlap) progression links
- $s_0 = B_0 \cup z_0$ is the initial state
  - $B_0$ is the initial belief base and $z_0$ is the initial node set of $I$
- $S_G = \{z_g | z_g \triangleright \triangleright_{tn} I\}$ is the goal state
  - $z_g$ is the terminal node set of $I$
Intention Interleaving Planning Formalism (Cont.)

A FPP problem of interleaving intentions $I = \{T_1, \cdots, T_m\}$ is a tuple

$$\Omega = \langle \Sigma, X, O, s_0, S_G \rangle$$

$$O = O^p \cup O^o$$

$\alpha^o = \{idx_b^1, \cdots, idx_e^k \rightarrow idx_e^1, \cdots, idx_e^k\} \in O^o$

in which $\alpha_i^p = \{idx_b^i \rightarrow idx_e^i\} \in O^p$

- $pre(\alpha^o) = pre(\alpha_1^p) \cup \cdots \cup pre(\alpha_k^p)$
- $del(\alpha^o) = del(\alpha_1^p) \cup \cdots \cup del(\alpha_k^p)$
- $add(\alpha^o) = add(\alpha_1^p) \cup \cdots \cup add(\alpha_k^p)$
A FPP problem of interleaving intentions \( I = \{T_1, \cdots, T_m\} \) is a tuple \( \Omega = \langle \Sigma, X, O, s_0, S_G \rangle \)

**Definition 1:** The result of applying a progression link \( \alpha \in O \) to a state \( s = B \cup z \) is described by the transition function \( f: 2^{\Sigma} \cup 2^X \times O \rightarrow 2^{\Sigma} \cup 2^X \) defined as follows:

\[
f(s, \alpha) = \begin{cases} 
(s \setminus \text{del}(\alpha)) \cup \text{add}(\alpha) & \text{if } s \models \text{pre}(\alpha) \\
\text{undefined} & \text{otherwise}
\end{cases}
\]

**Definition 2:** The result of applying a sequence of progression links to a state specification \( s \) is defined inductively:

\[
\text{Res}(s, \langle \alpha \rangle) = s \\
\text{Res}(s, \langle \alpha_0; \cdots; \alpha_n \rangle) = \text{Res}(f(s, \alpha_0), \langle \alpha_1; \cdots; \alpha_n \rangle)
\]

**Definition 3:** A sequence of progression links \( \Delta = \langle \alpha_0; \alpha_1; \cdots; \alpha_n \rangle \) is a solution to a FPP problem \( \Omega = \langle \Sigma, X, O, s_0, S_G \rangle \), denoted as \( \Delta = \text{sol}(\Omega) \), iff \( \text{Res}(s, \Delta) \models S_G \). We also say that \( \Delta \) is optimal if the sum of the size of the progression link \( \text{size}(\alpha_i) \) is maximum where \( i = 0, \cdots, n \).

**Theorem:** we have a maximal-merged trace \( \sigma^m \) of intention \( I = \{T_1, \cdots, T_m\} \) if and only if there exists an optimal solution \( \Delta \) to \( \Omega \).
Intention Interleaving Planning Formalism (Cont.)

A FPP problem of interleaving intentions $I = \{T_1, \ldots, T_m\}$ is a tuple $\Omega = \langle \Sigma, X, O, s_0, S_G \rangle$

```
Algorithm 1: Intention Interleaving Replanning

Input: Planning problem $\Omega = \langle \Sigma, X, O, s_0, S_G \rangle$

1. $\alpha_0; \ldots; \alpha_n \leftarrow sol(\Omega)$ /* FPP solution */
2. $i \leftarrow 0, \alpha \leftarrow \alpha_0, s \leftarrow s_0$ /* initialisation */
3. while $s \notin \Upsilon$ do
4.     if $f(s, \alpha) = \text{undefined}$ then
5.         $idx_b \leftarrow \text{BEGINNING-INDEX}(\alpha)$
6.         $G \leftarrow \text{BACKTRACK}(idx_b)$ /* backtrack */
7.         $s_0 \leftarrow \mathcal{B} \cup z \setminus \{idx_b\} \cup \{G\}$ /* modify state */
8.         $sol'(\Omega) \leftarrow \text{FPP}(\Sigma, X, O, s_0, S_G)$ /* replan */
9.         $\alpha_0; \ldots; \alpha_n \leftarrow sol'(\Omega)$
10.     end if
11. end while
12. EXECUTE $\alpha$
13. $s \leftarrow f(s, \alpha)$
14. $i \leftarrow i + 1$
15. $\alpha \leftarrow \alpha_{i+1}$

To adapt to the dynamic environment

line 5-7 instruct the procedures for failure backtracking and initial node state modification
```
Implementation

**Operator Files**
(containing progression links)

**Fact Files**
(containing initial/goal state description)

---

**Planning Domain Definition Language (PDDL)**

---

primitive progression links:

\[
(:\text{action} \ (idx_b \rightarrow P^T)) \\
:precondition \ (\text{and} \ idx_b \ \text{context}(P)) \\
:effect \ (\text{and} \ (\text{not} \ idx_b) \ P^T)) \\
(:\text{action} \ (idx_b \rightarrow a^{P,j,T}) \\
:precondition \ (\text{and} \ idx_b \ \psi(a^{P,j,T})) \\
:effect \ (\text{and} \ (\phi^-) \ \phi^+ (\text{not} \ idx_b) \ a^{P,j,T})) \\
(:\text{action} \ (idx_b \rightarrow G^{P,j,T}) \\
:precondition \ idx_b \\
:effect \ (\text{and} \ (\text{not} \ idx_b) \ G^{P,j,T}))
\]

overlap progression links:

\[
(:\text{action} \ \{\text{idx}_{x_1}^1, \ldots, \text{idx}_{x_n}^k \} \rightarrow \{\text{idx}_{x_1}^1, \ldots, \text{idx}_{x_n}^k \}) \\
:precondition \ (\text{and} \ \text{pre}(\alpha_1^k), \ldots, \text{pre}(\alpha_n^k)) \\
:effect \ (\text{and} \ \text{add}(\alpha_1^p) \ldots \text{add}(\alpha_n^p)) \\
\hspace{1cm} (\text{not} \ \text{del}(\alpha_1^p) \ldots (\text{not} \ \text{del}(\alpha_n^p)) \\
\hspace{1cm} (\text{increase} \ \text{(efficiency-utility)} \ \text{size}(\alpha^o)))
\]

---

\[(:\text{objects} \ \forall x \in X, \forall \text{BELIEF\_ATOMS} \in \Sigma)\]

\[(:\text{init} \ B_0, \forall T \in I, T(\bar{n}))\]

\[(:\text{goal} \ (\text{and} \ (\text{or} \ t_{n_{k_1}} \ldots t_{n_{k_1}}) \ldots (\text{or} \ t_{m_{k_1}} \ldots t_{m_{k_2}}))\]

---

declar all objects in the plan problem instance

initial belief base and the top-level goals of intentions

reach any terminal node of each intention
Evaluation: A Manufacturing Scenario

<table>
<thead>
<tr>
<th>Operation 1</th>
<th>Operation 2</th>
<th>Operation 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>block 1</td>
<td>twisting-drilling 10cm</td>
<td>reaming</td>
</tr>
<tr>
<td>block 2</td>
<td>twisting-drilling 15cm</td>
<td>reaming</td>
</tr>
<tr>
<td>block 3</td>
<td>twisting-drilling 20cm</td>
<td>reaming</td>
</tr>
<tr>
<td>block 4</td>
<td>twisting-drilling 25cm</td>
<td>reaming</td>
</tr>
</tbody>
</table>

Effectiveness Analysis of Approach

<table>
<thead>
<tr>
<th></th>
<th>2.1</th>
<th>2.2</th>
<th>3.1</th>
<th>3.2</th>
<th>3.3</th>
<th>4.1</th>
<th>4.2</th>
<th>4.3</th>
<th>4.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>17%</td>
<td>33%</td>
<td>11%</td>
<td>22%</td>
<td>33%</td>
<td>8%</td>
<td>17%</td>
<td>25%</td>
<td>33%</td>
</tr>
<tr>
<td>3</td>
<td>22%</td>
<td>44%</td>
<td>15%</td>
<td>30%</td>
<td>44%</td>
<td>11%</td>
<td>22%</td>
<td>33%</td>
<td>44%</td>
</tr>
<tr>
<td>4</td>
<td>25%</td>
<td>50%</td>
<td>17%</td>
<td>33%</td>
<td>50%</td>
<td>13%</td>
<td>25%</td>
<td>38%</td>
<td>50%</td>
</tr>
<tr>
<td>5</td>
<td>27%</td>
<td>53%</td>
<td>18%</td>
<td>36%</td>
<td>53%</td>
<td>13%</td>
<td>27%</td>
<td>40%</td>
<td>53%</td>
</tr>
<tr>
<td>6</td>
<td>28%</td>
<td>56%</td>
<td>19%</td>
<td>37%</td>
<td>56%</td>
<td>14%</td>
<td>28%</td>
<td>42%</td>
<td>56%</td>
</tr>
<tr>
<td>7</td>
<td>29%</td>
<td>57%</td>
<td>19%</td>
<td>38%</td>
<td>57%</td>
<td>14%</td>
<td>29%</td>
<td>43%</td>
<td>57%</td>
</tr>
<tr>
<td>8</td>
<td>29%</td>
<td>58%</td>
<td>19%</td>
<td>39%</td>
<td>58%</td>
<td>15%</td>
<td>29%</td>
<td>44%</td>
<td>58%</td>
</tr>
</tbody>
</table>

Details can be found in my github
https://github.com/Mengwei-Xu/manufacturing-evaluation
Summary:

1. Formalise an intention as AND/OR graph
2. Formalise the conflict-free execution trace of multiple intentions
3. Formalise the maximal-merged execution trace of multiple intentions
4. Define the concept of overlapping programs between different intentions
5. Both formally and practically compile the intention interleaving problem into a planning problem
6. Provide a preliminary evaluation of a planning-centric intention interleaving problem
Future Work:

1. A complete algorithm of computing overlap set of intentions
2. Further test the costs and benefits of our approach empirically in a wider range of applications
3. Investigate the collaboration between multi-BDI agents, e.g. how to discover and exploit collaboration opportunities