Items

There is a given set of items, $\mathcal{I} \subset \mathcal{C} \times \mathcal{T} \times \mathbb{N}$, $\mathcal{C}$ is a finite set of colors, $\mathcal{T}$ is a finite set of types and the third component is a unique identifier from $\mathbb{N}$.

Locations

The robot operates on a square grid of size $m \times n$. With $\mathcal{L} = \{1, \ldots, m\} \times \{1, \ldots, n\}$, location L is any subset of $\mathcal{L}$. A single element of $\mathcal{L}$ is called a *field*.

Specifications

Specifications are given in terms of temporal formulas (defined in the syntax section) and they use items and locations as their arguments.

Programs

Program is a sequence of specifications - they are executed one after another.

Model

Specifications are evaluated on traces of states. State is given as a pair (M, R). $M = (m)_{i,j}$ is a matrix of size $m \times n$ where each element corresponds to the field of the grid. $m_{i,j} = (\text{fieldItems}, \text{obs})$ with $\text{fieldItems} \subset \mathcal{I}$ and $\text{obs} \subset \{\text{free, wall, door}\}$. In other words, M gives information of what kind of field (i,j) is (free or one of the two kinds of obstacles - doors and walls) and which items are present at each field. Aforementioned R, on the other hand, tells about robot's current position and the items it carries. Formally, $R = (f, \text{robotItems})$, with $f = (i, j) \in \mathcal{L}$ and $\text{robotItems} \subset \mathcal{I}$ (with reasonable assumptions such as $m_{i,j}.\text{fieldItems} \cap m_{k,l}.\text{fieldItems} \neq \emptyset$ iff $i = k$ and $j = l$)

Items definitions

- $* \subset \mathcal{I} = \mathcal{I}$
- has color C $= \{i \in \mathcal{I} : i.\text{color} = C\}$
- has type T $= \{i \in \mathcal{I} : i.\text{type} = T\}$
- $\forall I_1, I_2 \subset \mathcal{I}, I_1 \& I_2 = I_1 \cap I_2$
- $\forall I_1, I_2 \subset \mathcal{I}, I_1 | I_2 = I_2 \cup I_2$

Locations definitions

Some of the locations that we refer to depend on the state. Therefore, the meaning of equality symbol (=) here is "interpreted as"

- $[[x_1, y_1], [x_2, y_2], \ldots, [x_n, y_n]] = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
- $[x, y] \equiv [[x, y]]$
- world $= \mathcal{L}$
- current$=\{\sigma(t).R.f\}$
- $L_1 + L_2 = L_1 \cup L_2$
- $L_1 * L_2 = L_2 \cap L_2$
- $L_1 - L_2 = L_1 \backslash L_2$
- L spread$(u, v) = \{(p_x + i, p_y + j) : i \in \{-u, \ldots, u\}, j \in \{-v, \ldots, v\}, p \in L\}$
- L spread$(s) \equiv L$ spread(s, s)
- L bounded spread$(u, v) = \bigcup_{\substack{i \in \{1,\ldots,u\} \\ j \in \{1,\ldots,v\}}} B_{i,j}$, where $f \in B_{i,j}$ iff $\sigma.M[f].\text{obs} = \text{free}$ and $\exists k \in \text{neighbours(f)}$ such that $k \in B_{i',j'}$, with $i' < i, j' < j$
- L with item $I = \{f \in L : \exists it \in \sigma.M[f].\text{fieldItems such that } it \in I\}$

Situations evaluation

Situations are specifications that can not be executed.

- for item description $I \subset \mathcal{I}$ and location $L \subset \mathcal{L}$ we define $\sigma, t \models I$ at L iff $I \cap \bigcup_{f \in L} \sigma(t).M[f].\text{fieldItems} \neq \emptyset$
- for item description $I \subset \mathcal{I}$ we define $\sigma, t \models I$ at self iff $I \cap \sigma(t).R.\text{robotItems} \neq \emptyset$
- for location $L \subset \mathcal{L}$ we define $\sigma, t \models \text{at}(L)$ iff $\sigma(t).R.f \in L$
- for a specification S we define $\sigma, t \models \text{possible}(S)$ iff $\sigma, t \models S$

Specifications evaluation

For a specification S we define its evaluation and the state that completes the specification, $\zeta(S)$.

- $\sigma, t \models$ visit (L_1) while avoiding L_2 iff $\exists k > t : \sigma, k \models (at)(L_1)$ and $\forall l \leq k$: $\neg(at)(L_2)$.
  $\zeta(\text{visit } (L)) = \sigma(k)$
- $\sigma, t \models$ visit periodically (L_1, p, N) while avoiding L_2 iff $\exists k$ such that $\sigma, t + k \models \text{at}(L)$ and $\sigma, t + k + p \cdot i \models \text{at}(L)$, for $i \in \{1, \ldots, N\}$ and $\forall l \leq t + k + p \cdot N$: $\sigma, l \models \neg(at)(L_2)$.
  $\zeta(\text{visit periodically } (L_1, p, N) \text{ while avoiding } L_2) = \sigma(t + k + p \cdot N)$
- $\sigma, t \models$ pick single I iff $\exists it \in I$ such that $it \in \sigma(t).M[\text{current}].\text{fieldItems}) \wedge (\sigma(t+1).R.\text{robotItems} = \sigma(t).R.\text{robotItems} \cup \{it\}) \wedge (\sigma(t+1).M[\text{current}].\text{fieldItems} = \sigma(t).M[\text{current}].\text{fieldItems} \backslash \{it\})$
  $\zeta(\text{pick single } I) = \sigma(t+1)$
- $\sigma, t \models$ pick all I iff $\sigma(t+1).M[\text{current}].\text{fieldItems} = (\sigma(t).M[\text{current}].\text{fieldItems} \backslash I) \wedge$
  $(\sigma(t+1).R.\text{robotItems} = \sigma(t).R.\text{robotItems} \cup (\sigma(t).M[\text{current}].\text{fieldItems} \cap I)$
  $\zeta(\text{pick all } I) = \sigma(t+1)$
- $\sigma, t \models$ drop single I iff $\exists it \in I$ such that $(it \in \sigma(t).R.\text{robotItems}) \wedge (\sigma(t+1).R.\text{robotItems} = \sigma(t).R.\text{robotItems} \backslash \{it\} \wedge (\sigma(t+1).M[\text{current}].\text{fieldItems} = \sigma(t).M[\text{current}].\text{fieldItems}) \cup \{it\}))$
  $\zeta(\text{drop single } I) = \sigma(t+1)$
- $\sigma, t \models$ drop all I iff $(\sigma(t+1).M[\text{current}].\text{fieldItems} = \sigma(t).M[\text{current}].\text{fieldItems} \cup (\sigma(t).R.\text{robotItems} \cap I))$
  $\wedge (\sigma(t+1).R.\text{robotItems} = \sigma(t).R.\text{robotItems} \backslash I)$
- $\sigma, t \models S$ within d for specification S iff $\exists k \leq d$ such that $\zeta(S) = \sigma(k)$.
  $\zeta(S \text{ within } d) = \sigma(k)$
- $\sigma, t \models$ no-op iff $\sigma(t) = \sigma(t+1)$
  $\zeta(\text{no-op}) = \sigma(t+1)$

Execution of specifications

If a specification S evaluates to true, then a path is synthesized that leads to state $\zeta(S)$ (and a user confirms whether he is happy with such a decision, given the simulation of the result). If it evaluates to false, the system stays in the same state and the user is warned that the spec is not realizable so he can change it.