

## SOAR - Architecture

- SOAR is a general problem solving architecture
- It uses heuristic search as solving paradigm
- The architecture is composed by five elements:
  - A production rules engine (stores the knowledge during the resolution and the results of the learning process)
  - Working memory (stores the current resolution)
  - Working memory manager (manipulates the working memory using the production rules)
  - Decision mechanism (allows to pick a resolution path in case of conflict)
  - A *chunking* mechanism (builds new rules from the resolution)

Notes

---

---

---

---

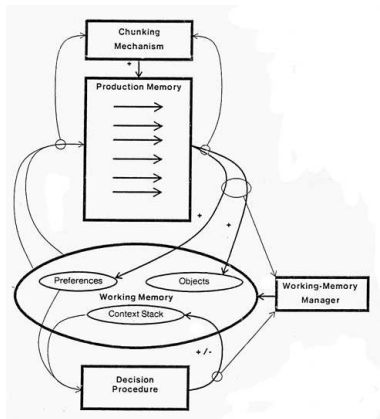
---

---

---

---

## SOAR - Architecture



Notes

---

---

---

---

---

---

---

---

## SOAR - Memory

- Stores different objects:
  - Context stack (represents the different points of resolution of the problem). Each element includes: The goal to solve, the space of problems (domain), the current state and the operator to apply
  - Working objects (Goals to accomplish, states achieved, ...)
  - Preferences about the resolution path (control information that allow to make decisions)

Notes

---

---

---

---

---

---

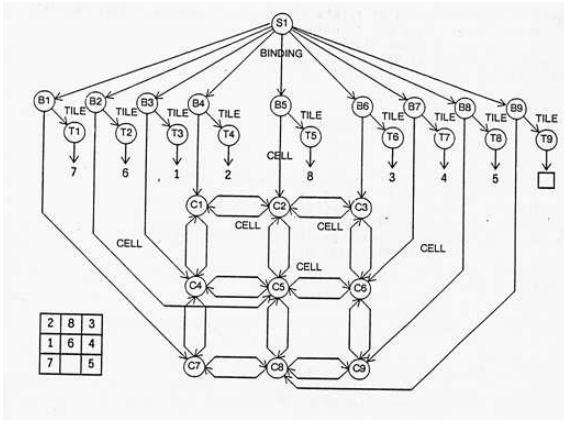
---

---





## SOAR - Example



Notes

---

---

---

---

---

---

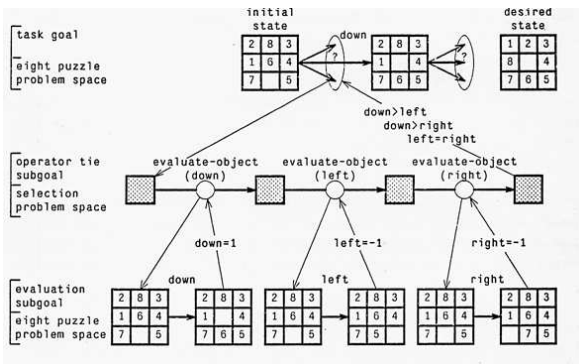
---

---

---

---

## SOAR - Example



Notes

---

---

---

---

---

---

---

---

---

---

## SOAR - Example

```

Cycle
0 G: G1 [Solve the eight puzzle]
1 P: P1 [Eight-Puzzle]
2 S: S1
3 G: G2 [Tile impasse, operators (O1[down] O2[left] O3[right])]
4 P: P2 [Selection]
5 S: SS1
6 G: G3 [evaluate-object(O1[down])]
7
eval*select-role-operator
  (goal G2 *operator O4)           :wm elements tested to
  (operator O4 *name evaluate-object *desired D1 :establish the context
  *role operator *superoperator O1           :in which operator O1[down]
  *superproblem-space P1 *superstate S1)
-->
8 P: P1 [Eight-Puzzle]
9 S: S1
10 O: O1 [down]
create-new-state
  (problem-space P1 *name eight-puzzle) :wm elements tested to
  (operator O1 *name move-tile radjacent-cell C1) :apply operator that moves
  (state S1 *binding B1 *binding B2)           :the tile in C1 into the
  (binding B1 *tile T1 *cell C2)             :cell with the blank (C2)
  (tile T1 *name blank)                     :T1 is the blank
  (binding B2 *tile T2 *cell C1)             :T2 is the tile in cell C1
-->
11 S: S2
eval*state-plus-one
  (problem-space P1 *name eight-puzzle) :wm elements tested to
  (operator O4 *name evaluate-object       :create evaluation for
  *desired D1 *revaluation E1)           :state based on detecting
  (desired D1 *binding DB1)              :that the operator
  (binding DB1 *cell C2 *tile T2)         :has moved a tile into
  (cell C2 *cell C1)                    :its desired position
-->
  (evaluation E1 *value 1)               :the result/action
12 O: O5 [evaluate-object(O2[left])]
  
```

Notes

---

---

---

---

---

---

---

---

---

---

Notes

```
(sp p0038
  • (goal <G2> †operator <O4>)
  • (operator <O4> †name evaluate-object †role operator
    †superproblem-space <P1> †superstate <S1>
    †superoperator <O1> †evaluation <E1> †desired <D1>)
  • (problem-space <P1> †name eight-puzzle)
  • (operator <O1> †adjacent-cell <C1>)
  • (state <S1> †binding <B1> †binding { <> <B1> <B2> })
  • (binding <B1> †tile <T1> †cell <C2>)
  • (tile <T1> †name blank)
  • (binding <B2> †cell { <> <C2> <C1> } †tile { <> <T1> <T2> })
  • (cell <C2> †cell <C1>)
  • (desired <D1> †binding <DB1>)
  • (binding <DB1> †cell <C2> †tile <T2>)
-->
  (evaluation <E1> †value 1))
```

Notes

Notes