

Grundlagen der Künstlichen Intelligenz

2 Rationale Agenten

Rationale Agenten und ihre Umgebungen

Volker Steinhage

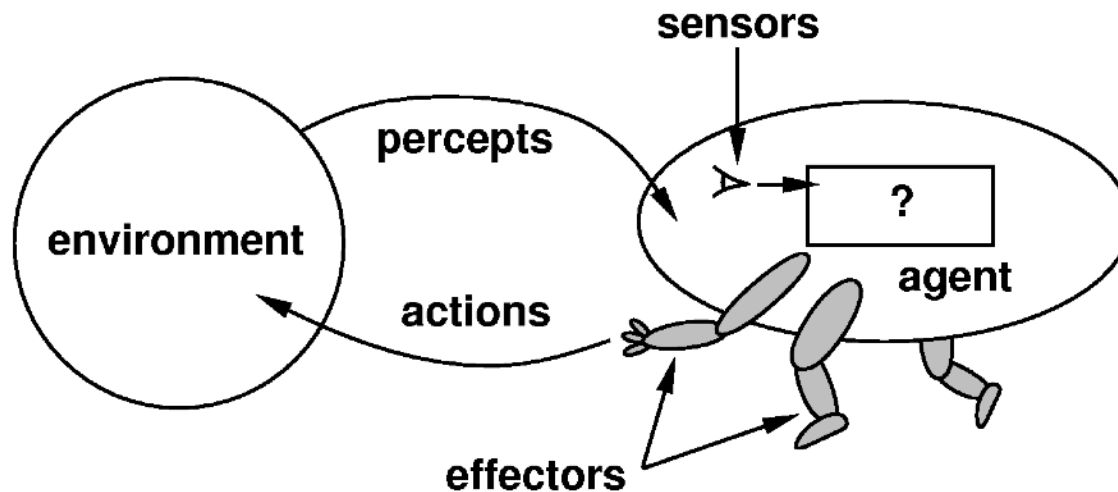
Inhalt

- Was ist ein Agent?
- Was ist ein *rationaler* Agent?
- Welche *Struktur* hat ein rationaler Agent?
- Welche *Klassen* von Agenten gibt es?
- Welche *Klassen* von Umgebungen gibt es?

Agenten

Aus der ersten Vorlesung: Agenten

- nehmen durch *Sensoren* ihre *Umgebung (Umwelt)* wahr (\rightarrow *Perzepte*)
- manipulieren ihre *Umgebung* mit Hilfe ihrer *Effektoren (Aktuatoren)* (\rightarrow *Aktionen*)

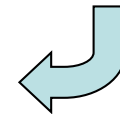
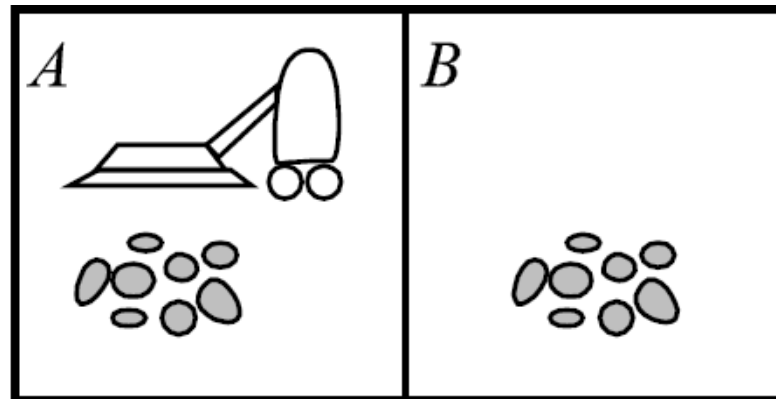


- Was sind nun genauer:
- Umgebung,
 - Sensoren und Perzepte,
 - Aktuatoren und Aktionen?

Agenten: Beispiel 1

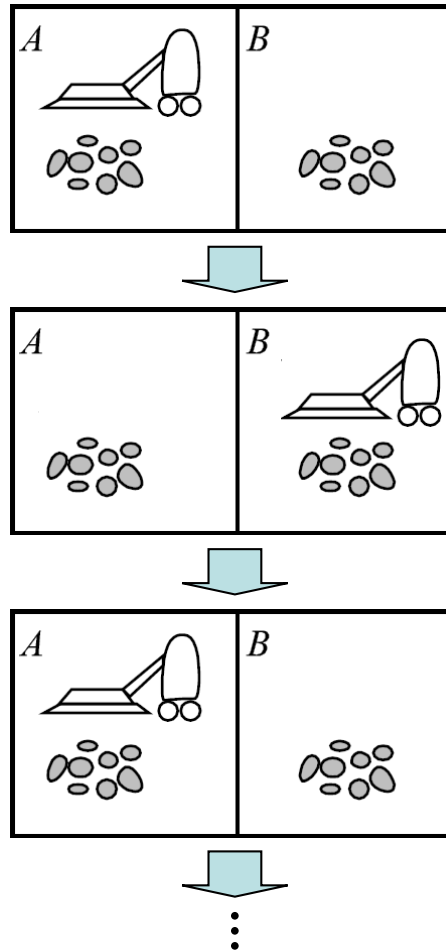
Der abstrakte Staubsauger-Agent

- **Umgebung:** Räume A und B – ggf. jeweils mit Schmutz
- **Sensoren:** Positionssensor und Schmutzsensor
- **Perzepte:** Positionsangabe (hier: binäre Raumangabe A oder B),
Schmutzangabe (hier: binäre Erkennung von Schmutz (j/n))
- **Aktuatoren:** Fahrwerk, Sauger
- **Aktionen:** nach links/rechts fahren, saugen



Agenten: Beispiel 1

Der Staubsauger-Agent könnte nun beliebig lange hin- und herfahren und dabei ... nichts tun!



Einem solch ziellosen Handeln haben wir in der ersten Vorlesung das Konzept des **rationalen Agenten** gegenüber gestellt \leadsto **Zielerfüllung!**

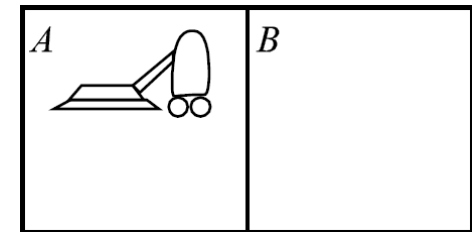
Rationale Agenten ...

... machen das „Richtige“ zur Zielerfüllung!

~ die Beurteilung erfordert **objektive Erfolgs-** oder **Leistungskriterien!**

Für den **abstrakten Staubsauger-Agenten**:

beide Räume A und B sind ohne Schmutz



Für eine **technische Umsetzung** des Staubsauger-Agenten sind noch genauere bzw. zusätzliche Kriterien möglich wie:

- gesäuberte m² pro Stunde
- Reinheitsgrad
- Stromverbrauch
- Geräuschemission
- Sicherheit

Gibt es eine allgemeine Systematik zur Spezifikation des Leistungsumfangs eines Agenten

?

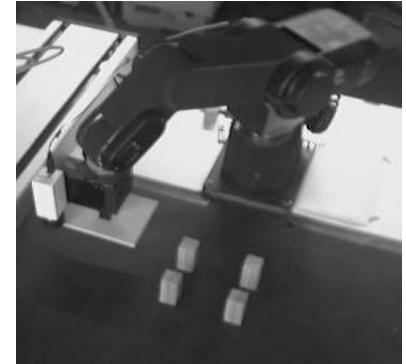
PEAS-Spezifikation

- Die **Spezifikation von rationalen Agententypen** erfolgt über vier Kriterien:
 - **Umgebung**, in welcher der Agent agiert
 - **Sensoren** des Agenten
 - **Aktuatoren** des Agenten
 - **Leistungskriterien**, die der Agent zu erfüllen hat
- Diese Spezifikation wird nach den engl. Entsprechungen der Kriterien - Performance, Environment, Actuators, Sensors - als **PEAS-Spezifikation** bezeichnet.

Beispiele für PEAS-Spezifikation

Einfacher Roboterarm-Agent

- **Umgebung:** Arbeitsplatte, Bauklötze
 - **Sensoren:** Kamera, taktile Sensoren
 - **Aktuatoren:** Roboterarm, Greifer
 - **Leistungskriterium:** Bauklötze in best. Endposition bringen
-



Automatischer Taxifahrer-Agent

- **Umgebung:** Straßen, anderer Verkehr, Fußgänger, Fahrgäste
- **Sensoren:** Kamera, Sonar, Tachometer, GPS, Kilometerzähler, Motorsensoren, Tastatur/Mikrofon
- **Aktuatoren:** Steuerrad, Gas, Bremse, Hupe, Blinker, Anzeige
- **Leistungskriterium:** Sicher, schnell, StVO-gemäß, angenehme Fahrweise, maximaler Gewinn



Weitere Beispiele rationaler Agenten:

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	healthy patient, minimize costs, lawsuits	patient, hospital, staff	display questions, tests, diagnoses, treatments, referrals	keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	correct image categorization	downlink from orbiting satellite	display categorization of scene	color pixel arrays
Part-picking robot	percentage of parts in correct bins	conveyor belt with parts, bins	jointed arm and hand	camera, joint angle sensors
Refinery controller	maximize purity, yield safety	refinery, operators	valves, pumps, heaters, displays	temperature, pressure, chemical sensors
Interactive English tutor	maximize student's score on test	set of students, testing agency	display exercises, suggestions, corrections	keyboard entry



Von allen Kriterien ist die gegebene **Umgebung** am wenigsten zu ändern.

~ Bewertung von Umgebungen bzgl. zu erwartender Schwierigkeiten ist wünschenswert.

Die Umgebung rationaler Agenten

- Vollständig vs. teilweise beobachtbar:

Sind *alle* relevanten Aspekte der Welt von den Sensoren beobachtbar?

- Deterministisch vs. stochastisch vs. strategisch:

Deterministisch: nächster Weltzustand hängt allein vom aktuellen Zustand und der ausgeführten Aktion ab. Strategische Umgebung: deterministisch ist bis auf Aktionen anderer Agenten. Sonst: stochastisch.

- Episodisch vs. sequentiell:

Kann die Qualität einer Aktion einfach innerhalb einer *Episode* (*Perzept + Aktion*) bewertet werden oder muss die vorherige und/oder zukünftige Entwicklung für die Qualitätsbewertung auch berücksichtigt werden?

- Statisch vs. dynamisch vs. semidynamisch:

Kann sich die Welt ändern, während der Agent reflektiert?
Semidynamisch: Welt ist statisch, aber ihre Bewertung ist dynamisch.

- Diskret vs. kontinuierlich:

Ist die Welt diskret (Schachspielen) oder nicht (beweglicher Roboter)?

- Einzel- vs. Multi-Agenten:

Müssen andere Entitäten der Umgebung als Agenten betrachtet werden?

Es kann kooperative und kompetitive Szenarien und Mischformen geben.

Beispiele für Umgebungen

Task	Observability	Next state	Evaluation of Actions	Environment	State space	#Agents
Crossword puzzle	fully	deterministic	sequential	static	discrete	single
Chess with a clock	fully	strategic	sequential	semi	discrete	multi
poker	partially	stochastic	sequential	static	discrete	multi
backgammon	fully	stochastic	sequential	static	discrete	multi
taxi driving	partially	stochastic	sequential	dynamic	continuous	multi
medical diagnosis	partially	stochastic	sequential	dynamic	continuous	single
image analysis	fully	deterministic	episodic	semi	continuous	single
part-picking robot	partially	stochastic	episodic	dynamic	continuous	single
refinery controller	partially	stochastic	sequential	dynamic	continuous	single
Interactive English tutor	partially	stochastic	sequential	dynamic	discrete	multi

Idealer rationaler Agent

Bisherige Betrachtung der Agenten über ihr Verhalten, also die **äußere Wechselwirkung** in ihrer Umgebung, aufgrund ihrer Wahrnehmungsfolgen und Aktionen.

Demnach ist ein idealer rationaler Agent wie folgt beschreibbar als

- Agent, der für alle möglichen Wahrnehmungssequenzen und bei gegebenem Weltwissen die Aktion wählt, welche die erwartete Leistung maximiert.
- Beschrieben durch eine *Agentenfunktion*:

Rationaler_Agent: Wahrnehmungssequenz \times Weltwissen \rightarrow Aktion

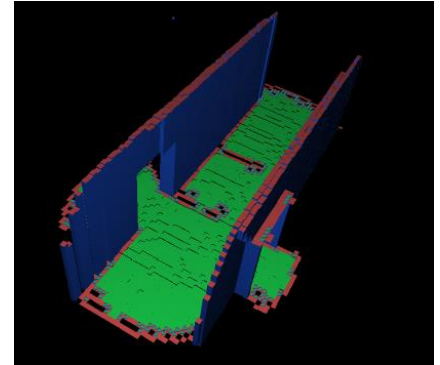
Ab jetzt Betrachtung der inneren Struktur, d.h.

- des **Aufbaus** von Agenten bzw.
- der **Implementierung** einer Agentenfunktion

Die Struktur rationaler Agenten

Realisierung der *Agentenfunktion* durch ein

- *Agentenprogramm*, das auf einer
(im Folgenden durch *Pseudocodes* dargestellt)
- *Architektur* ausgeführt wird, die auch
die Schnittstellen zur Umwelt realisiert
(Sensoren/Perzepte, Effektoren/Aktionen)



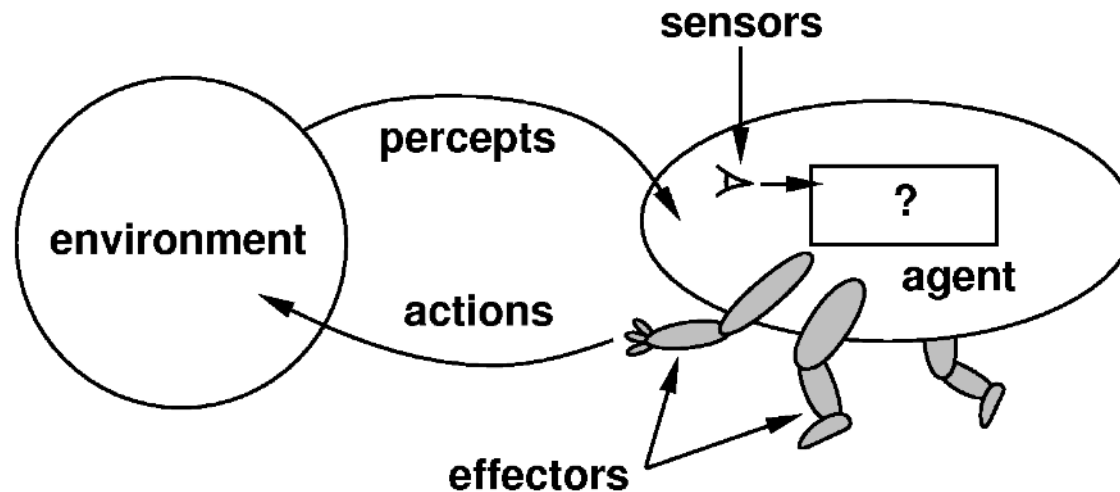
→ *Agentenstruktur* = *Agenten*architektur + *Agenten*programm

Agentenprogramme

Agentenprogramme im Pseudocode zeigen i.A. dieselbe Signatur:

```
function AGENT(percept) returns action
```

Dies entspricht einer Rückkopplungsschleife des Agenten:



Das einfachste Design: Tabellengesteuerte Agenten (1)

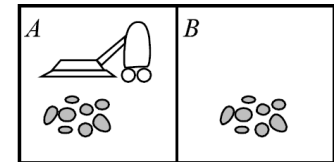
Die **einfachste Kodierung eines Agentenprogramms** ist die explizite tabellarische Repräsentation der Agentenfunktion.

Diese ordnet jeder möglichen Wahrnehmungssequenz eine Aktion zu:

```
function TABLE-DRIVEN-AGENT(percept) returns action  
  static: percepts, a sequence, initially empty  
           table, a table, indexed by percept sequences, initially fully specified  
  
  append percept to the end of percepts  
  action  $\leftarrow$  LOOKUP(percepts, table)  
  return action
```

Das einfachste Design: Tabellengesteuerte Agenten (2)

Die Tabelle für das Agentenprogramm des Staubsauger-Agenten:



A sauber	Nach rechts
A schmutzig	Saugen
B sauber	Nach links
B schmutzig	Saugen
A sauber , A sauber	Nach rechts
A sauber, A schmutzig	Saugen
...	...
A sauber, A sauber, A sauber	Nach rechts
A sauber, A sauber, A schmutzig	Saugen
...	...

Annahme: der Staubsauger-Agent habe eine „Lebensdauer“ von nur 10 *Wahrnehmungssequenz-Aktions-Paaren*, in der er

- beide Räume A und B überwachen und pflegen soll und
- beide Räume A und B auch wieder verschmutzen können

~ die Tabelle hat bei 4 möglichen Perzepten bereits über eine Million Einträge!

Das einfachste Design: Tabellengesteuerte Agenten (3)

Allg. ergibt sich die Tabellengröße bei einer Menge P von möglichen Perzepten und einer Lebensdauer von T Wahrnehmungssequenz-Aktions-Paaren zu:

$$\sum_{t=1}^T |P|^t.$$

Erfolgen die Perzepte des automatisierten Taxis über Kamerabilder mit ca. 27 MB/sec (30 Bilder á 640 x 480 Pixel mit 24 Bit Farbtiefe pro Sekunde) so hat die Tabelle eine Größe von ca. $10^{250.000.000.000}$ Einträgen für eine Stunde Fahrt.

Selbst für Schach, einem winzigen und wohl definierten Fragment der realen Welt, hat die Tabelle eine Größe von mind. 10^{150} Einträgen.

Solche Tabellen müssten also zunächst explizit **aufgestellt** oder durch Training **gelernt** werden, um dann nach jeder Wahrnehmung **angefragt** zu werden.

Fünf Basisklassen von Agentenprogrammen

Außerdem: es handelt sich auch *nicht* um eine *intelligent* kodierte Agentenfunktion, da jedes Paar von Wahrnehmungssequenz und Aktion explizit notiert und damit „*auswendig gelernt*“ wird.

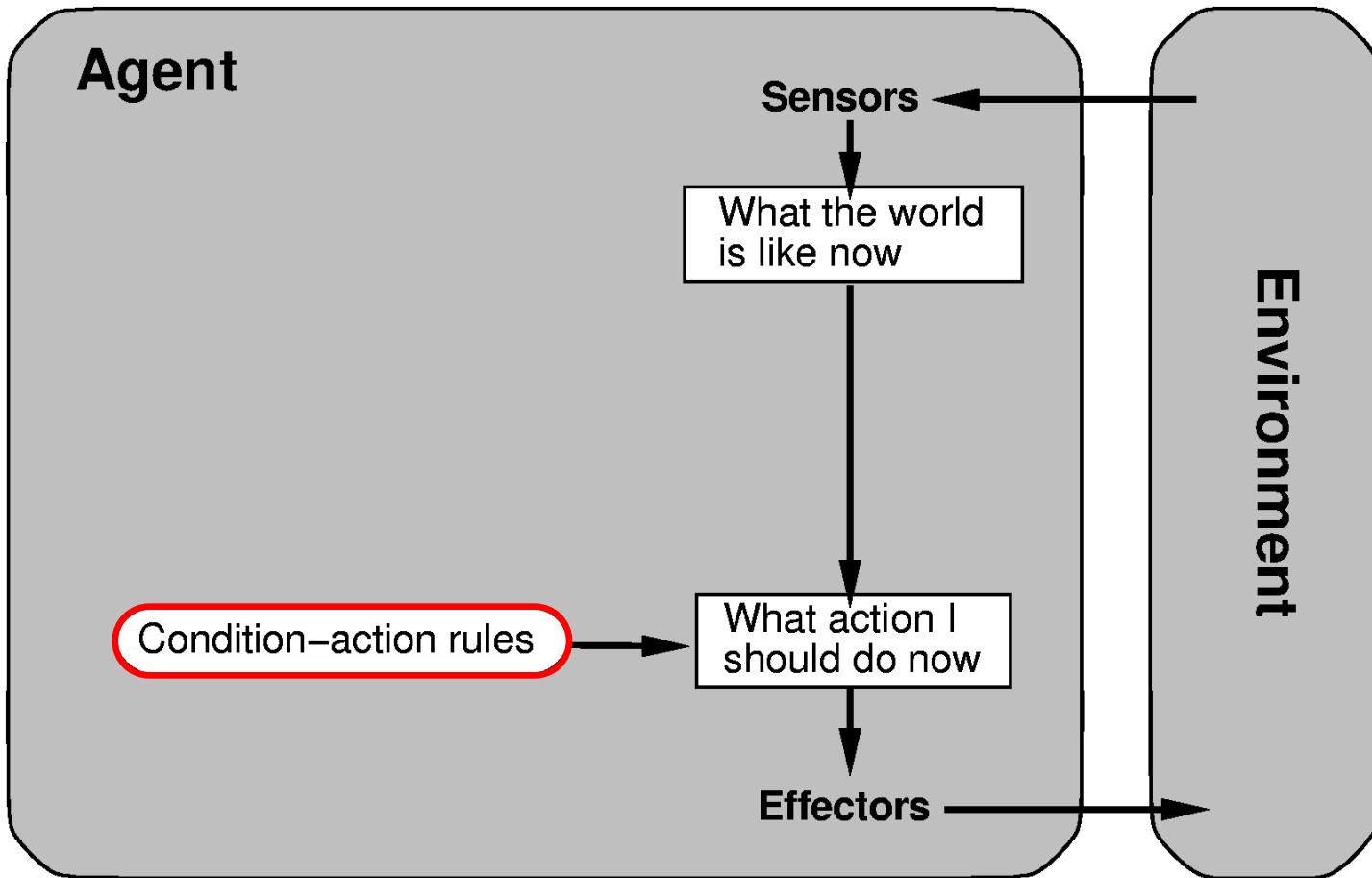
Es erfolgt so keine Generalisierung, die wir von intelligenten Systemen erwarten.

Wir erwarten eher Lösungen, in denen die Agentenfunktion durch ein kompakt und damit algorithmisch eleganter kodiertes Agentenprogramm umgesetzt wird.

~ Fünf Basisklassen von Agentenprogrammen für intelligente Systeme:

- einfache reflexive Agenten
- reflexive Agenten mit Weltmodell
- zielorientierte Agenten
- nutzorientierte Agenten
- lernende Agenten

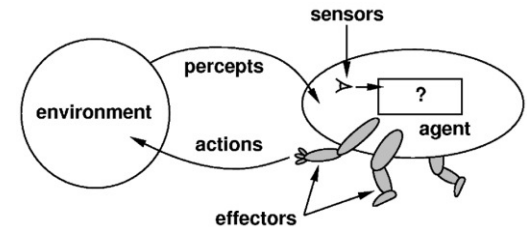
Einfache reflexive Agenten



Die einfachste Umsetzung: Reflektion und Reaktion alleine aufgrund des aktuellen Perzeptes unter Vernachlässigung der Perzepthistorie.

Programm des einfachen reflexiven Agenten

```
function SIMPLE-REFLEX-AGENT(percept) returns action  
  static: rules, a set of condition-action rules  
  
  state  $\leftarrow$  INTERPRET-INPUT(percept)  
  rule  $\leftarrow$  RULE-MATCH(state, rules)  
  action  $\leftarrow$  RULE-ACTION[rule]  
  return action
```

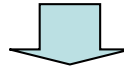


Hier *state* = direkte Interpretation nur aufgrund des aktuellen Perzeptes

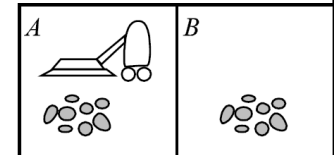
- Geeigneter Ansatz für *vollständig beobachtbare Umwelt*: Das aktuelle Perzept umfasst die relevante Information für die Auswahl der Aktion.
- Probleme bei Verdeckungen, Verschattungen, Unschärfen, also allg. bei nicht wahrgenommener bzw. wahrnehmbarer Information.
- Sinnvoll: für Lösung einfacher Aufgaben und Auslösen von Reflexen
Bspl.: Bremslichter des vorausfahrenden KFZ leuchten auf → Bremsen!

Programm des einfachen reflexiven Staubsauger-Agenten (1)

```
function SIMPLE-REFLEX-AGENT(percept) returns action  
  static: rules, a set of condition-action rules  
  
  state  $\leftarrow$  INTERPRET-INPUT(percept)  
  rule  $\leftarrow$  RULE-MATCH(state, rules)  
  action  $\leftarrow$  RULE-ACTION[rule]  
  return action
```



```
function REFLEX-VACUUM-AGENT([location, status]) returns action  
  
  if status = Dirty then return Suck  
  else if location = A then return Right  
  else if location = B then return Left
```



status und *location* sind für den abstrakten Staubsauger-Agenten bereits interpretierte Perzepte und hier in ihren Kombinationen *states**. Der rechte Teil der drei *if*- bzw. *else-if*-Regeln spezifiziert sofort die *action*.

* Bei digitalen Bildern als Perzepten würde INTERPRET-INPUT erst durch Bildanalyse eine Interpretation wie etwa *location=A* und *status=Dirty* aus den Bildern ableiten müssen.

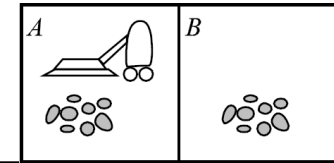
Programm des einfachen reflexiven Staubsauger-Agenten (2)

```
function REFLEX-VACUUM-AGENT([location, status]) returns action
```

```
if status = Dirty then return Suck
```

```
else if location = A then return Right
```

```
else if location = B then return Left
```



Das reflexive Staubsauger-Agentenprogramm ist sehr viel kompakter als das tabellengesteuerte Staubsauger-Agentenprogramm.

Die kompaktere Darstellung basiert i. W. auf der Reduktion des Wahrnehmungsverlaufs:

Reduktion der zu betrachtenden Fälle von $\sum_{t=1, \dots, T} 4^t$ auf 4.

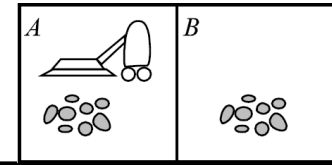
Programm des einfachen reflexiven Staubsauger-Agenten (3)

```
function REFLEX-VACUUM-AGENT([location, status]) returns action
```

```
  if status = Dirty then return Suck
```

```
  else if location = A then return Right
```

```
  else if location = B then return Left
```



Aber:

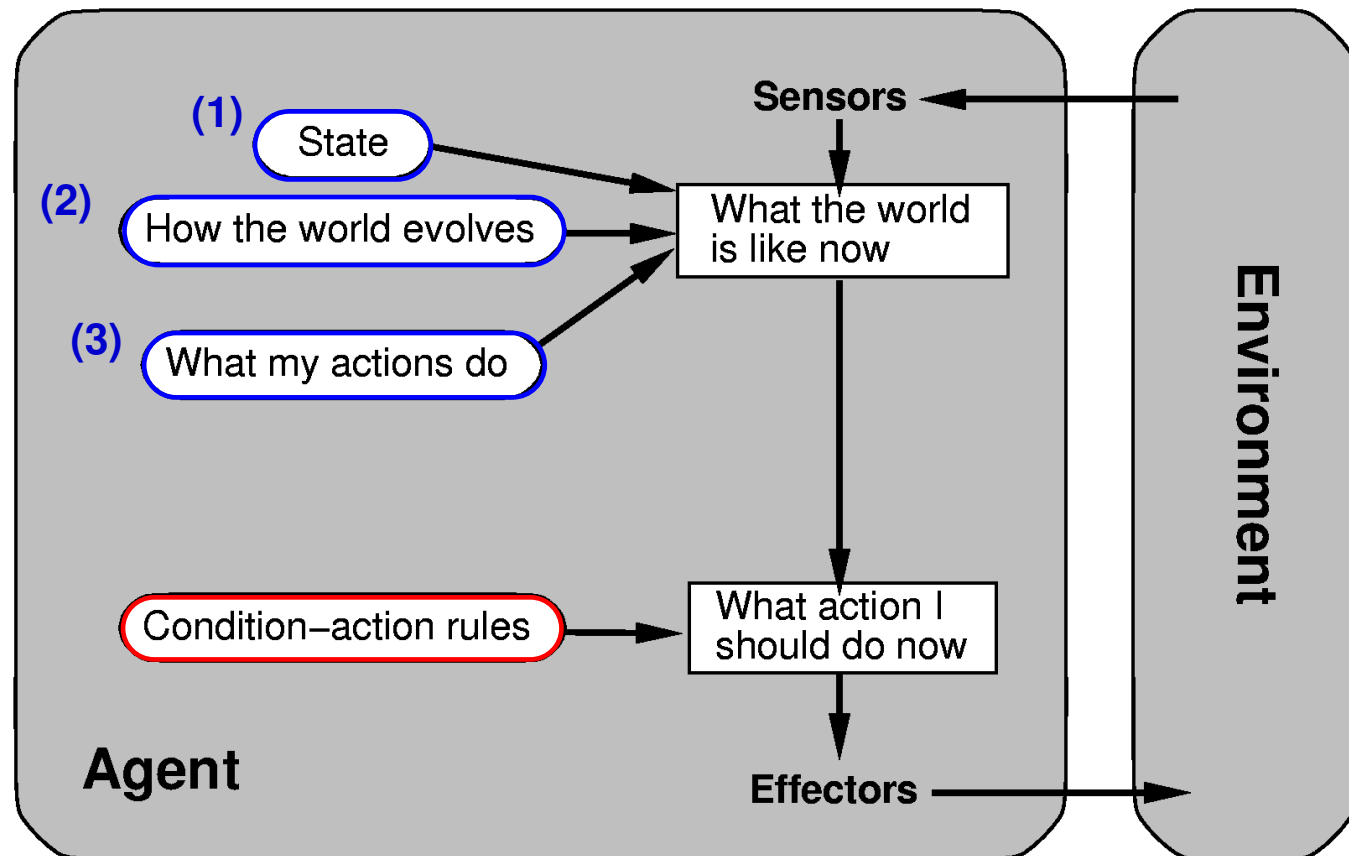
- der einfache reflexive **Staubsauger-Agent** weiß nicht, **wann er fertig ist**, weil er sich nicht merkt, wenn Räume A und B sauber sind.
- ein einfacher reflexiver **Taxifahrer-Agent** kann sich nicht merken, dass die Durchfahrt durch Straße A eben wegen einer Baustelle scheiterte und versucht ggf. später nochmals vergeblich Straße A zu durchfahren.

Reflexiver Agent mit Weltmodell

Ein *internes Umweltmodell* bestimmt neben dem aktuellen Perzept die Auswahl von Aktionen.

Das Umweltmodell umfasst:

- (1) Zustandmodell
- (2) Änderungsmodell
der Umwelt
- (3) Wechselwirkungs-
modell von Agent
mit Umwelt.



Programm des reflexiven Agenten mit Weltmodell (1)

function REFLEX-AGENT-WITH-STATE(*percept*) **returns** an action

static: *state*, a description of the current world state

rules, a set of condition-action rules

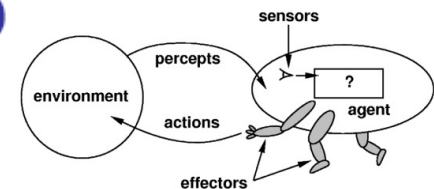
action, the most recent action, initially none

state \leftarrow **UPDATE-STATE**(*state*, *action*, *percept*)

rule \leftarrow RULE-MATCH(*state*, *rules*)

action \leftarrow RULE-ACTION[*rule*]

return *action*



Hier **neue Zustandsbeschreibung** aufgrund von

- **aktuellem Zustand +**
- **Perzept +**
- **ausgeführter Aktion**

Das Weltmodell ist umfassender als eine Perzepthistorie, da im Modell auch nicht beobachtbare Zusammenhänge durch Änderungs- und Wechselwirkungsmodell einfließen.

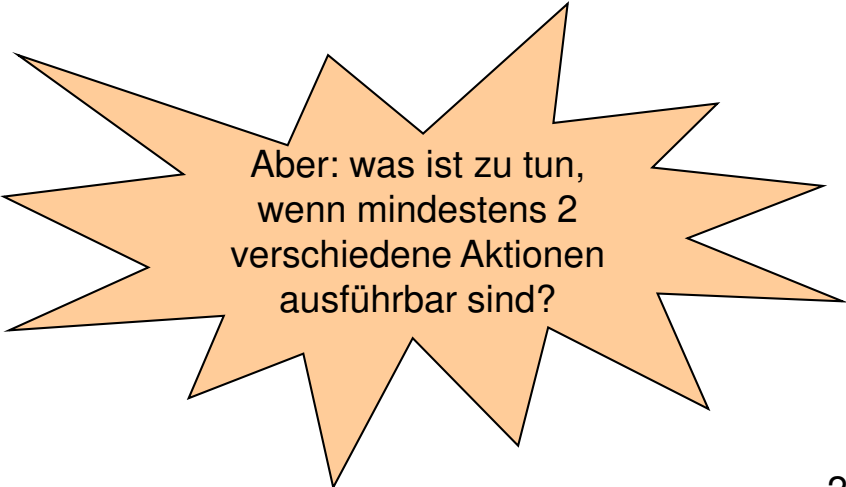
Programm des reflexiven Agenten mit Weltmodell (2)

```
function REFLEX-AGENT-WITH-STATE(percept) returns an action
  static: state, a description of the current world state
           rules, a set of condition-action rules
           action, the most recent action, initially none

  state  $\leftarrow$  UPDATE-STATE(state, action, percept)
  rule  $\leftarrow$  RULE-MATCH(state, rules)
  action  $\leftarrow$  RULE-ACTION[rule]
  return action
```

Bspl. 1: Staubsauger-Agent merkt sich Ergebnis des Saugens in einem Raum und weiß nach Saugen des anderen Raumes, dass aufgehört werden kann.

Bspl. 2: Fahrbahnwechsel erfordert Berücksichtigung der *Zusammenhänge* zwischen Fahrzeugen auf mehreren Fahrbahnen in Vorfeld und Rückfeld!



Aber: was ist zu tun, wenn mindestens 2 verschiedene Aktionen ausführbar sind?

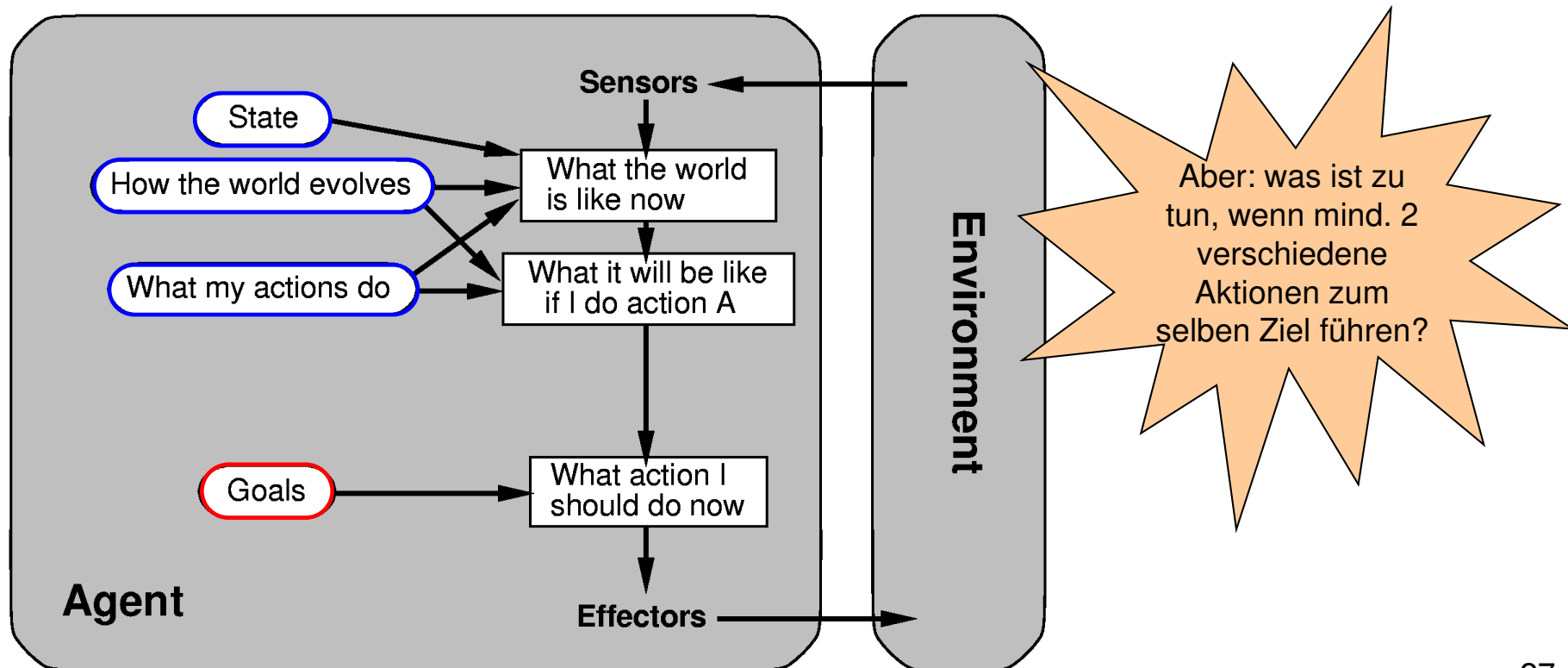
Zielorientierte Agenten (2)

Weltmodell und Perzepte sind für die Aktionsauswahl dann nicht ausreichend, **wenn die richtige Aktion von explizit vorgegebenen Zielen abhängt.**

Bspl.: Zustand erlaubt an Kreuzung das Fahren nach links, rechts und geradeaus

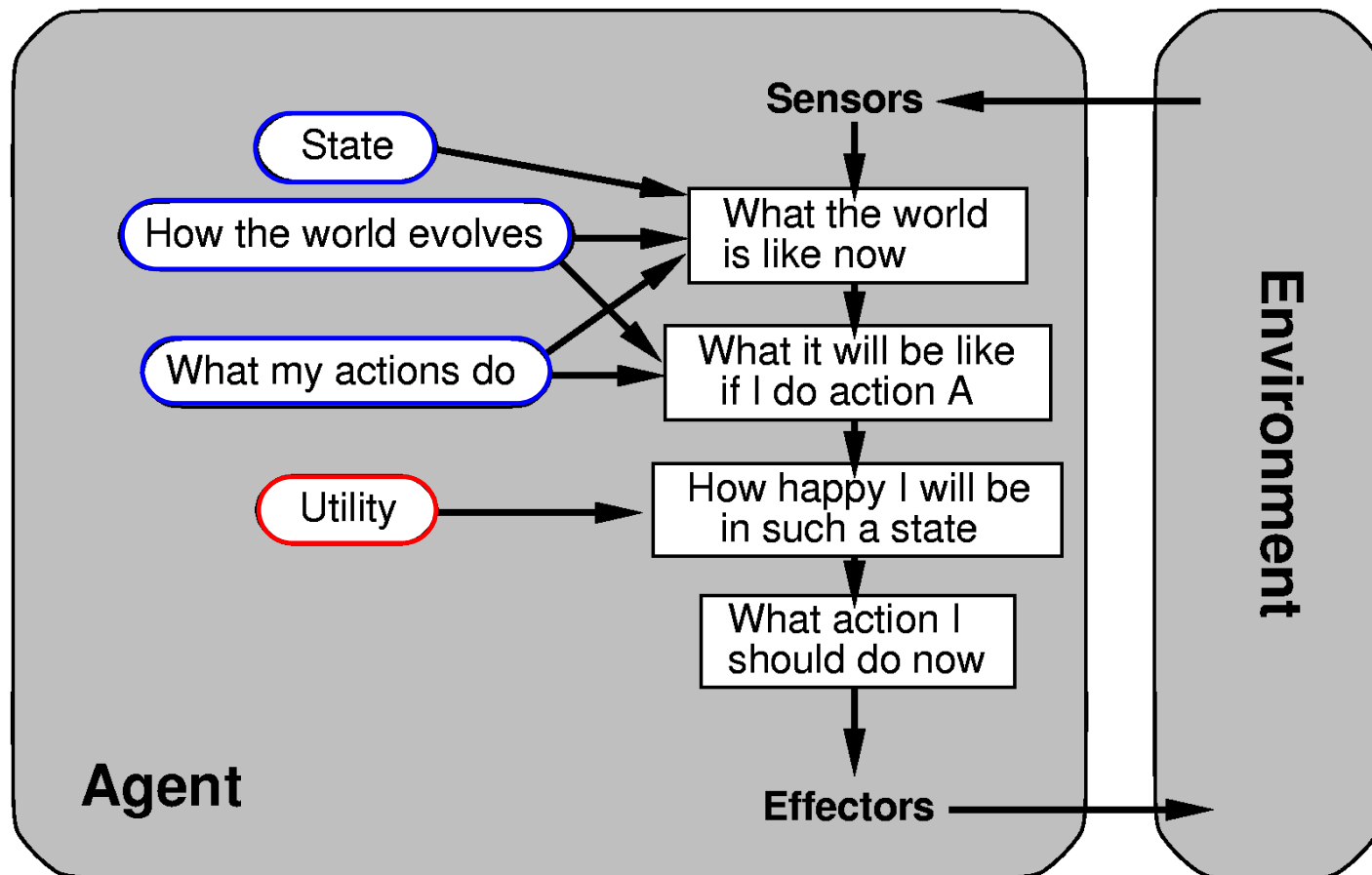
→ **Alleine das Ziel kann hier entscheiden!**

→ Explizite Repräsentation von Zielen und deren Berücksichtigung bei Aktionswahl.



Nutzenorientierte Agenten (1)

Meist gibt es mehrere Aktionsfolgen, die zu einem Ziel führen. Dann kann der **Nutzen** (*Utility*) des erreichten Zustands herangezogen werden, um eine Auswahl zu treffen.



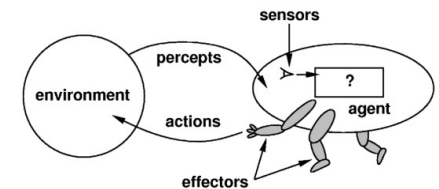
Nutzenorientierte Agenten (3)

Explizite Nutzenfunktionen sind umfassender als Zielrepräsentation, da komplexere Zusammenhänge einfließen können:

- 1) *In Konflikt stehende Teilziele* sind verrechenbar!

Bspl.: Teilziel 1: Möglichst schnell zum Bahnhof.

Teilziel 2: Sicher fahren.



- 2) *Unsichere Teilziele* sind verrechenbar, indem sowohl ihre Wichtigkeit als auch ihre Erzielbarkeit verrechenbar sind.

Bspl.: Ziel 1: Den Mantel aus der Reinigung zu holen, ist zwar wichtiger, aber wegen des nahen Ladenschlusses unwahrscheinlicher erfolgreich durchführbar.

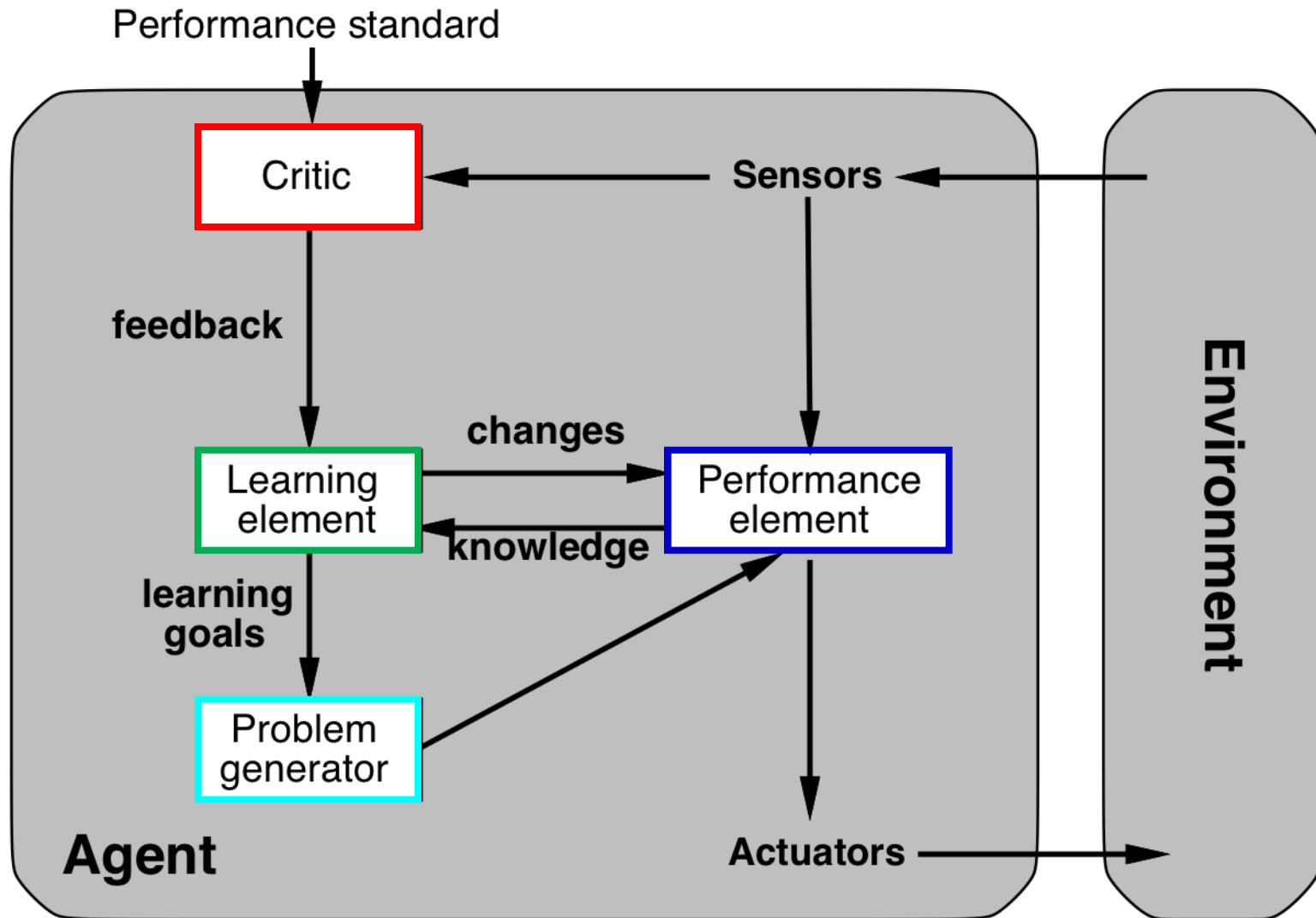
Ziel 2: An der Tankstelle Brötchen zu kaufen, ist zwar weniger wichtig, aber zeitlich unkritischer durchführbar.

Lernende Agenten (1)

Der lernende Agent evaluiert sich quasi selbst, um bessere Performanz zu erzielen. Dazu werden **vier Komponenten** eingesetzt:

1. **Performanzkomponente** (*Performance element*): eines der bisher beschriebenen Agentenkonzepte.
2. **Bewertung** (*Critic*) der bisherigen Performanzergebnisse mit vorgegebenen *Performanzstandards*.
3. **Lernkomponente** (*Learning element*) zur *Generierung von Änderungsvorschlägen* für die Performanzkomponente aufgrund der Bewertung der bisherigen Performanzergebnisse
4. **Problemgenerator** zur *Auswahl von Aktionen, die der Exploration dienen*. (I.A. lernen wir z.B. von grenzwertigen und/oder unerprobten Aktionen mehr als von sicheren Aktionen. Solche Aktionen sind aber im allg. Ausführungsmodus zu vermeiden, wenn hohe Sicherheit und hohe Performanz gefragt sind.)

Lernende Agenten (2)



Zusammenfassung

- Ein *Agent* besteht aus einer *Agentenarchitektur* und einem *Agentenprogramm*.
- Ein Agent *nimmt seine Umgebung wahr* und *agiert*.
- Ein *idealer rationaler Agent* führt die Aktionen aus, die für gegebene Wahrnehmung und gegebenes Weltwissen die *erwartete Leistung maximieren*.
- Ein *Agentenprogramm* bildet Perzepte auf Aktionen ab.
- Es existieren fünf Basisklassen von Agentenprogrammen:
 - *einfache reflexive* Agenten entscheiden allein gemäß ihrer Perzepte,
 - *modellbasierte* Agenten entscheiden gemäß ihres Weltmodells,
 - *zielorientierte* Agenten versuchen, gegebene Ziele zu erreichen,
 - *nutzenorientierte* Agenten maximieren ihre Bewertung,
 - *lernende* Agenten verbessern sich durch Vergleich mit Standards selbst.
- Unterschiedliche *Umgebungen* erfordern verschiedene Agentenkonzepte. Unzugängliche, nicht-episodische, dynamische und kontinuierliche Umgebungen sind die schwierigsten.

Ausblick

Umsetzung von Problemlösung bzw.
Zielerreichung durch Suchverfahren

