به نام خدا

آشنایی با عامل ها و انواع معماری آنها

جواد ایزدی

آبان ۱۳۹۳

کد کارگاه **WR1403**

فهرست مطالب

- تعریف عامل
- انواع عامل
- انواع معماری عامل

تعریف عامل

- تعریف ۱(Brustloni ۱۹۹۱) : عامل سیستمی خود مختار و هدفمند در دنیای واقعی است.
- این تعریف نشان می دهد که عاملها باید قادر به انجام کارها و زندگی و عمل در دنیای واقعی باشند به طوری که اعمال هدفداری را انجام دهند.

تعریف عامل (ادامه)

- تعریف ۲(Shoham ۱۹۹۳) : یک عامل، موجودیتی است که وضعیت آن شامل اجزای ذهنی نظیر باورها، انتخابها و تعهدات است. این اجزا به صورتی دقیق تعریف می شوند و تقریبا معادل همتاهای خود در باور عمومی هستند. از این منظر، عامل بودن در ذهن برنامه نویس است: آنچه که یک قطعه سخت افزاری یا نرم افزاری را به عامل تبدیل می کند، این واقعیت است که استفاده کننده تصمیم گرفته است آن را با توجه به این قالب ذهنی تحلیل و کنترل نماید.

تعریف عامل (ادامه)

- تعریف۳(Smith ۱۹۹۴) : عامل را به صورت یک موجودیت نرم افزاری دائمی، که به یک منظور خاص تخصیص داده شده است تعریف کنیم: "دائمی" بودن، عاملها را از زیربرنامه ها متمایز می سازد؛ عاملها ایده های خاص خود را در مورد نحوه به انجام رساندن کارها ، و نیز فهرست کاری خاص خود را دارند. "خاص منظوره" بودن، عاملها را از نرم افزارهای بزرگ چندمنظوره متمایز می سازد؛ عاملها معمولا بسیار کوچکتر هستند.

تعریف عامل (ادامه)

- تعریف۴(Pattie Maes ۱۹۹۵) : عاملهای خودکار، سیستمهای محاسباتی هستند که در محیط محاسباتی پیچیده ای حضور داشته، به صورت خودکار در محیط خود حس و عمل می کنند، و بدین وسیله مجموعه ای از اهداف یا وظایف را که برای انجام آن طراحی شده اند، به انجام می رسانند

تعریف عامل (ادامه)

- تعریف ۵ (Russel and Norvig ۱۹۹۵) : عامل هر چیزی است که بتوان آن را به صورت دریافت کننده ادراکات از محیط از طریق حسگرها و عمل بر روی محیط از طریق عملگرها مشاهده نمود.
- اسکلت یک عامل:

```
Function SKELETON-AGENT (percepts) returns action
        Static Memory, the agent's memory of the world
        Memory ← UPDATE-MEMORY (memory, percept)
        Action ← CHOOSE-BEST-ACTION (memory)
        Memory ← UPDATE-MEMORY (memory, action)
Return action
```

تعریف عامل (ادامه)

- تعریف ۶ (Etzioni and Weld ۱۹۹۵) : یک عامل باید دارای مشخصات زیر باشد:
  - خودمختاری
    - عامل، رفتار خود را از راه های زیر کنترل می کند:
      - هدفگرایی
      - هماهنگی
      - انعطاف پذیری
      - خودآغازی
  - تداوم زمانی
  - شخصیت
  - توانایی برقراری ارتباط
  - تطبیق پذیری
  - قابلیت تحرک

تعریف عامل (ادامه)

- تعریف ۷ (Hayes-Roth ۱۹۹۵) : سیستمهای هوشمند بطور پیوسته در حال انجام سه عمل می باشند: ادراک شرایط پویا در محیط، عمل به منظور تاثیر بر شرایط در محیط و استدلال برای تفسیر ادراکات، حل مسایل، استنتاج و تعیین اعمال.

تعریف عامل (ادامه)

- تعریف ۸ (Wooldridge and Jennings ۱۹۹۵) : عامل یک سیستم کامپیوتری است که ویژگی های زیر را دارا می باشد:
- خود مختاری: عامل ها بدون مداخله مستقیم انسانها یا دیگران عمل می کنند و نوعی کنترل بر اعمال و حالات داخلی خود دارند.
- اجتماعی بودن: عاملها با عاملهای دیگر (و شاید انسانها) از طریق نوعی زبان ارتباطی فعل و انفعال می کنند.
- قابلیت واکنشی: عاملها محیط خود را ادراک می کنند و به تغییراتی که در آن رخ می دهد در زمان معینی پاسخ می دهند.
- کنش گرایی: عامها در پاسخ به محیط خود رفتار هدف گرا نشان می دهند.

تعریف عامل (ادامه)

- تعریف ۹ (Franklin and Graesser ۱۹۹۷) : عاملهای هوشمند موجودیت های نرم افزاری ای هستند که مجموعه ای از اعمال را در ورای یک کاربر یا یک برنامه دیگر با درجه ای از استقلال یا خودمختاری انجام می دهند و در انجام آن از دانش یا نمایشی از اهداف و خواسته های کاربر استفاده می کنند.
- تعریف ۱۰ (Weiss ۱۹۹۹) : عامل یک واحد محاسباتی مانند یک برنامه نرم افزاری یا یک روبات است که می توان آن را به این صورت در نظر گرفت که در محیط خود ادراک و عمل می کند و خودمختار است از آن جهت که رفتار آن حداقل بصورت جزیی به تجربه اش بستگی دارد.

پارامترهای موجود در تعاریف عامل

پارامتر / تعریف	خود مختاری	موقعیت گرا	واکنشی	کنش گرا	یادگیری	صداقت	پایداری	اجتماعی	هدف گرا	استدلال	سازگاری	حرکت	نوع دوستی	نمایندگی	شایستگی	احساس مسئولیت	قدرت استدلال	عقلانیت
۱	*								*									
۲									*									
۳	*								*									
۴	*	*	*						*									
۵		*	*															
۶	*							*	*		*	*						
۷		*	*							*							*	
۸	*		*	*				*										
۹	*								*									
۱۰	*	*	*															

خصوصیات یک عامل

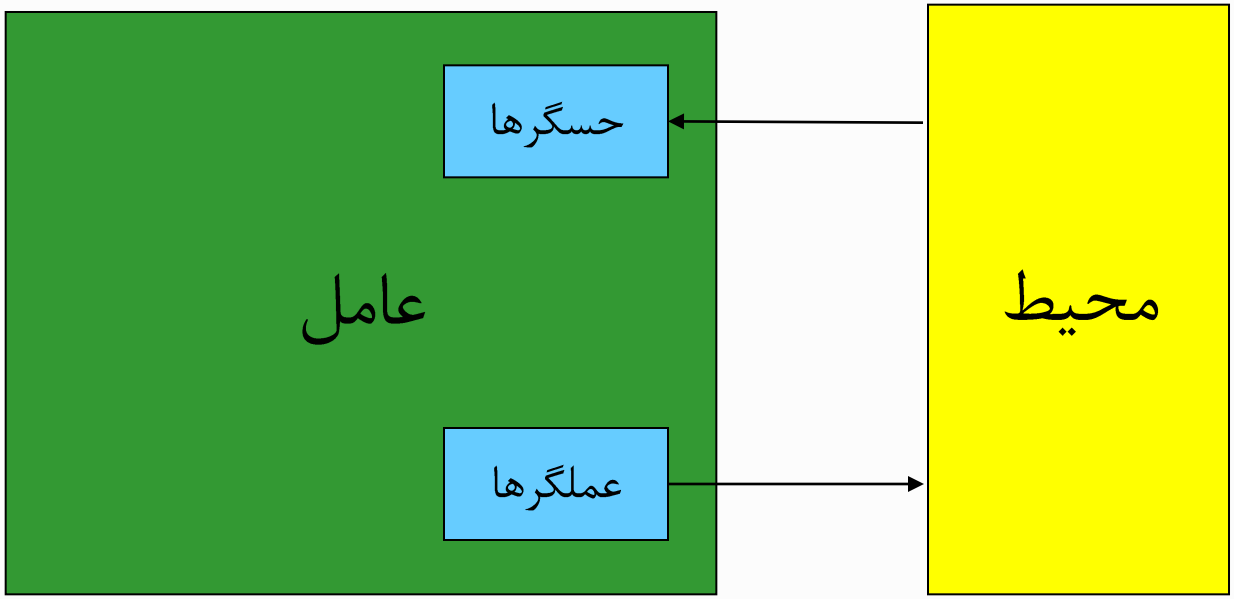
- واکنشی
- خودمختاری
- هدفگرا
- دارای ارتباط
- یادگیری
- ...

پارامترهای کلیدی در تعریف عامل

- از جدول قبلی نتیجه می گیریم که هر قطعه کدی که ویژگی های زیر را نداشته باشد عامل نیست :
  - خود مختاری
  - موقعیت گرا
  - واکنشی
  - هدف گرا

تعریف ما از عامل

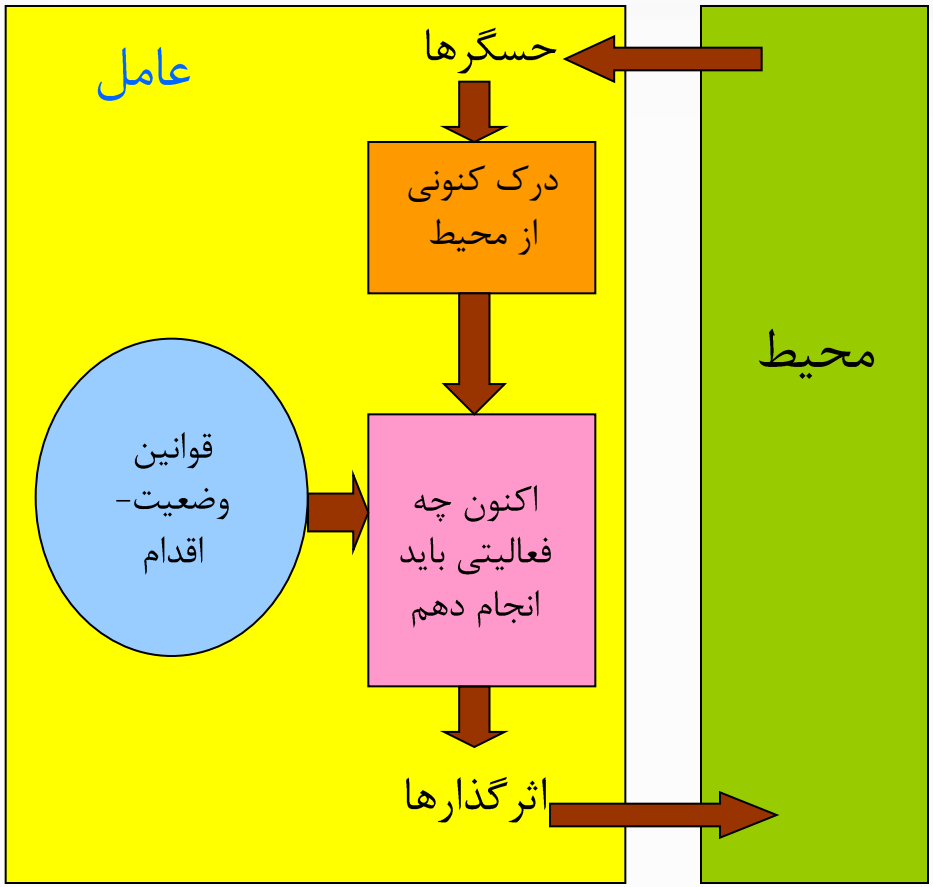
- عامل یک **سیستم کامپیوتری** کپسوله شده است که در **محیط قرار گرفته** و توانایی **رفتار انعطاف پذیر و خودکار** در محیط را برای **دستیابی به اهدافی که برای آنها طراحی شده** است، دارا می باشد.



انواع عامل ها

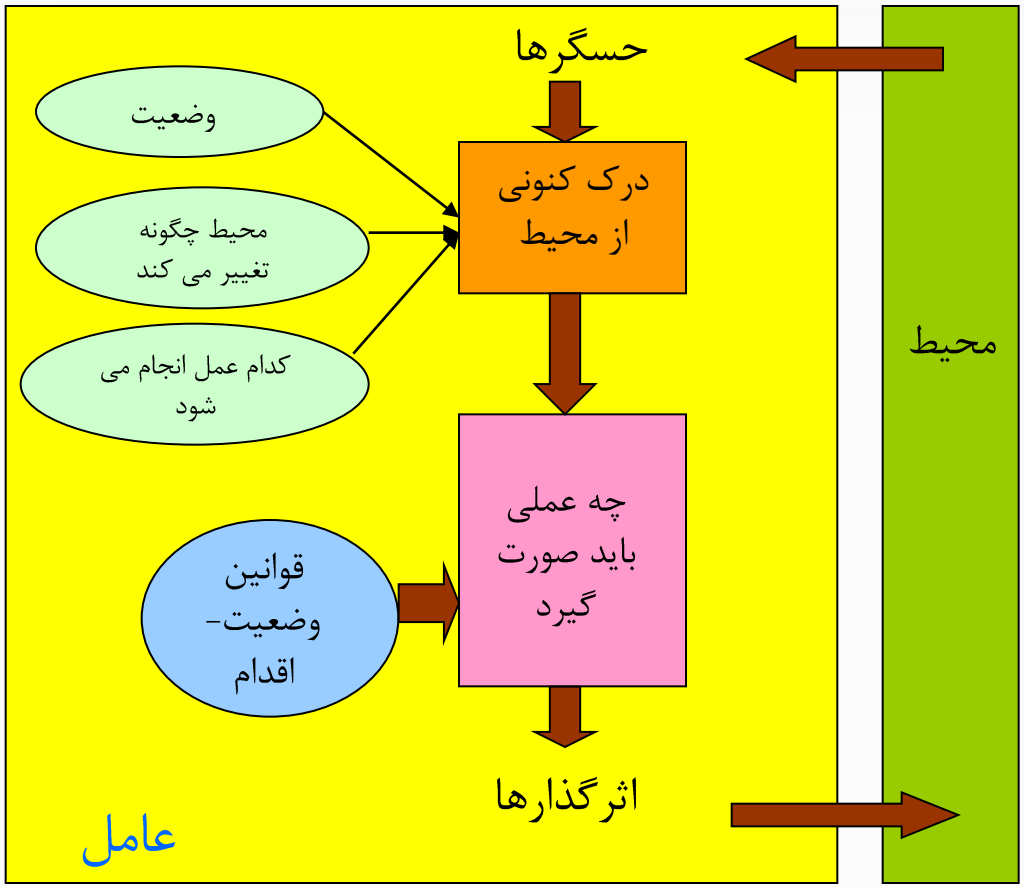
- عامل های واکنشی ساده
- عامل های وابسته به محیط
- عامل های هدف گرا
- عامل های سودمند

عامل های واکنشی ساده



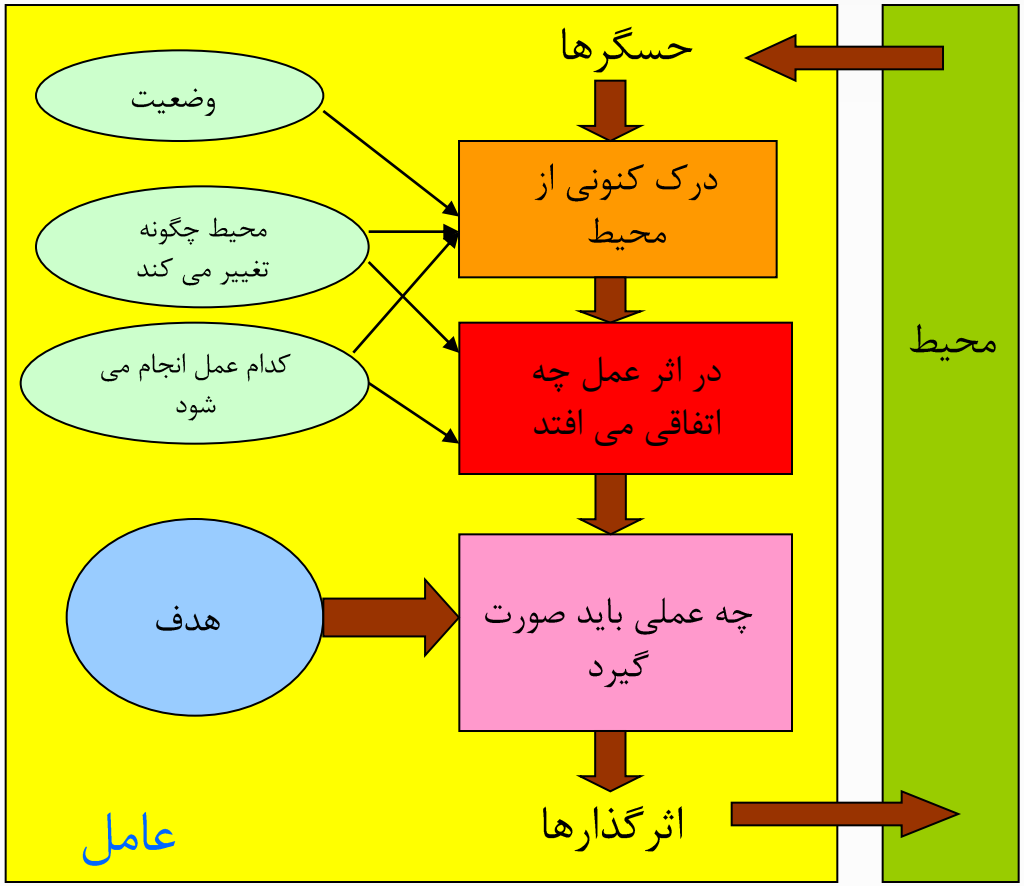
- در این نوع از عامل ها محیط از طریق حسگرها شناخته شده و فرایند تصمیم گیری با استفاده از پردازش وضعیت داخلی و اطلاعات پس زمینه انجام می گیرد.
- این نوع از عامل ها قادر به برنامه ریزی آینده نمی باشند و تنها محدود به آنچه انجام می دهند هستند.
- فعالیت هایشان توسط درک کنونی تعیین می شود و دانشی از آنچه انجام می دهند ندارند همچنان که اطلاعی از آنچه که سعی در رسیدن به آن دارند در دست ندارند.

عامل های وابسته به محیط



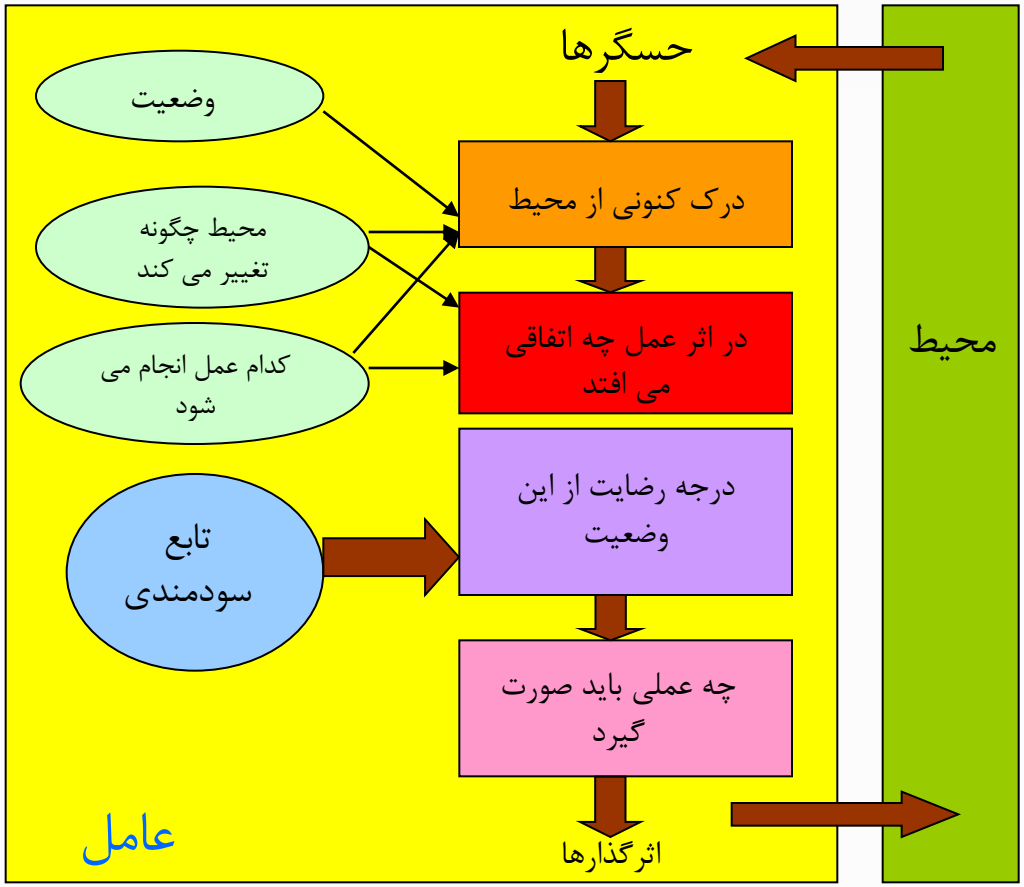
- این عامل ها با جمع آوری اطلاعات درباره چگونگی تغییرات محیط و اینکه رفتار آنها چه تاثیری بر روی محیط می گذارد می توانند عملکرد مناسبی بر روی محیط داشته باشند.
- با توجه به این که حسگرها نمی توانند دسترسی کامل به وضعیت دنیا بوجود آورند لازم است در برخی از وضعیت های داخلی عامل دستکاری صورت گیرد.
- به منظور به هنگام سازی اطلاعات وضعیت داخلی همزمان با گذر زمان نیازمند دو نوع دانش کد شده در برنامه عامل هستیم. (به Module جدید نیاز است)
  - اول این که نیاز داریم برخی اطلاعات درباره چگونگی تغییرات جهان مستقل از عامل ها را داشته باشیم.
  - دوم این که نیازمند اطلاعاتی درباره اعمال خود عامل هستیم که بر روی محیط اثر می گذارد.

عامل های هدف گرا



- دانستن در مورد وضعیت محیط همواره برای تصمیم گیری کافی نمی باشد. برای مثال تاکسی در یک چهارراه می تواند به سمت چپ یا راست تغییر مسیر دهد. تصمیم صحیح به مقصد بستگی دارد.
- برنامه عامل می تواند اطلاعات هدف را با اطلاعات درباره اعمال ممکن ترکیب نموده تا عمل مناسب را برای رسیدن به هدف انتخاب کند.
- این نوع عامل ها بسیار انعطاف پذیر می باشند. به سادگی با تعیین یک هدف تازه می توان عامل هدف گرا را به رفتار تازه رساند.

عامل های سودمند



- اهداف به تنهایی برای تولید رفتار با کیفیت بالا مناسب نیستند به عنوان مثال دنباله های زیادی از اعمال وجود دارند که قادرند تاکسی را به مقصد برسانند اما برخی سریع تر، امن تر یا ارزان تر هستند. به عبارت دیگر اهداف همواره یک معیار خام بین وضعیت رضایت یا عدم رضایت فراهم می کنند.
- سودمندی تابعی است که یک وضعیت را به عددی حقیقی نگاشت می کندکه درجه رضایت را تشریح می کند.
مشخصات کامل تابع سودمندی امکان تصمیم گیری های منطقی را برای هر نوع هدف فراهم می کند.

انواع معماری عاملها

- چهار نوع معماری برای چهار گروه از عامل ها مطرح است:
  - عامل های منطقی/ نمادین
  - عامل های واکنشی
  - عامل های BDI
  - عامل های ترکیبی و چند لایه

تاریخچه

- در سالهای ۱۹۵۶ تا ۱۹۸۵ تقریبا تمام عامل هایی که در هوش مصنوعی طراحی می شدند بر مبنای استدلال بودند و از استدلال منطقی در این گونه عامل ها برای تصمیم گیری استفاده شده است.
- در سال ۱۹۸۵ با توجه به مشکلات استدلال نمادین عامل های واکنشی مطرح شدند.
- از سال ۱۹۹۰ به بعد تکنیک های معماری ترکیبی ارائه شدند که سعی در ترکیب بهترین معماری های استدلالی و واکنشی را داشته اند.

عامل های منطقی/ نمادین

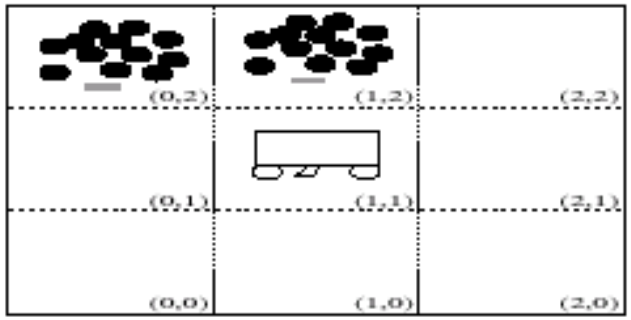
- ساده ترین روش برای ساخت عامل ها این است که آنها را نوع خاصی از سیستم های مبتنی بر دانش بدانیم.
- این الگو هوش نمادین نامیده می شود .
- یک معماری منطقی معماری ای است که :
  - در آن عامل شامل مدلی نمادین از دنیای واقعی است که بطور صریح بیان می شود.
  - تصمیمات آن بر مبنای استدلال نمادین یا منطق صورت می گیرد.

عامل های منطقی/ نمادین (ادامه)

- مشکلات عمده عامل های منطقی
  - تعریف تابع نگاشت محیط به ادراک به شکل نمادین مشکل است. به عبارت دیگر تعریف خصوصیات یک محیط پویا و واقعی در قالب مجموعه ای از قوانین استنتاج دشوار است.
  - زمان لازم برای تصمیم گیری این عامل ها قابل کنترل نبوده و احتمال دارد تابع تصمیم گیری هرگز پایان نگیرد.

عامل های منطقی/ نمادین (ادامه)

- مثال: دنیای جاروبرقی
  - هدف روبات روبات جستجوی محیط، کشف آشغال و جارو کردن آن است.



$In(x,y)$	agent is at (x,y)
$Dirt(x,y)$	there is dirt at (x,y)
$Facing(d)$	the agent is facing direction d

$$In(x,y) \wedge Dirt(x,y) \longrightarrow Do(suck)$$
$$In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) \longrightarrow Do(forward)$$
$$In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) \longrightarrow Do(forward)$$
$$In(0,2) \wedge Facing(north) \wedge \neg Dirt(0,2) \longrightarrow Do(turn)$$
$$In(0,2) \wedge Facing(east) \longrightarrow Do(forward)$$

عامل های منطقی/ نمادین (ادامه)

- عامل دنیای جاروبرقی
  - از سه گزاره در این مثال استفاده می کنیم:
    - *In(x,y)* عامل در خانه *(x,y)* قرار دارد
    - *Dirt(x,y)* در خانه *(x,y)* آشغال وجود دارد
    - *Facing(d)* جهت عامل به سمت *d* است که *d* می تواند شمال، جنوب، شرق یا غرب باشد.
  - اعمال ممکن
    - *Ac ={turn,forward,suck}* که در آن *turn* به معنی گردش به راست می باشد.

عامل های منطقی/ نمادین (ادامه)

- قوانین برای تعیین این که چه عملی انجام شود

$$In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) \longrightarrow Do(forward)$$
$$In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) \longrightarrow Do(forward)$$
$$In(0,2) \wedge Facing(north) \wedge \neg Dirt(0,2) \longrightarrow Do(turn)$$
$$In(0,2) \wedge Facing(east) \longrightarrow Do(forward)$$

و ...

- عامل با استفاده از این قوانین و با شروع از خانه (۰و۰) شروع به برداشتن آشغال می کند.

عامل های واکنشی

- مسائل حل نشده فراوانی در رابطه با هوش مصنوعی نمادین وجود دارد.
- این مشکلات باعث شد که تعدادی از محققین در پی یافتن جانشینی برای آن باشند و سه دیدگاه مطرح شد:
  - دیدگاه اول به رد بازنمایی نمادین و تصمیم گیری براساس ساختار آن پرداخته است
  - دیدگاه دوم این ایده را دارد که رفتار هوشمند عامل ناشی از تعامل با محیط است و بطور مستقل قابل تعریف نیست.
  - در دیدگاه سوم رفتارهای هوشمند متشکل و منتج از رفتارهای ساده تر دانسته شده است. این دیدگاه عامل را مبتنی بر رفتار می داند.

عامل های واکنشی (ادامه)

- Rodney Brooks معماری جزء به کل را مطرح کرد که در آن دو خصوصیت مطرح است:
  - اول این که تصمیم گیری عامل از طریق مجموعه ای از رفتارهای مستقل صورت می گیرد و بر محیط اثر می گذارد. یعنی درک از محیط براساس اثر رفتار به عنوان ورودی نگاشتی از حالت به عمل می دهد.
  - دوم این که تعدادی رفتار بطور همزمان می توانند اجرا شوند.

عامل های واکنشی (ادامه)

- در این معماری رفتارها بصورت لایه های مختلفی ارائه می شوند که
  - هر رفتار یک ساختار شبیه قانون دارد:

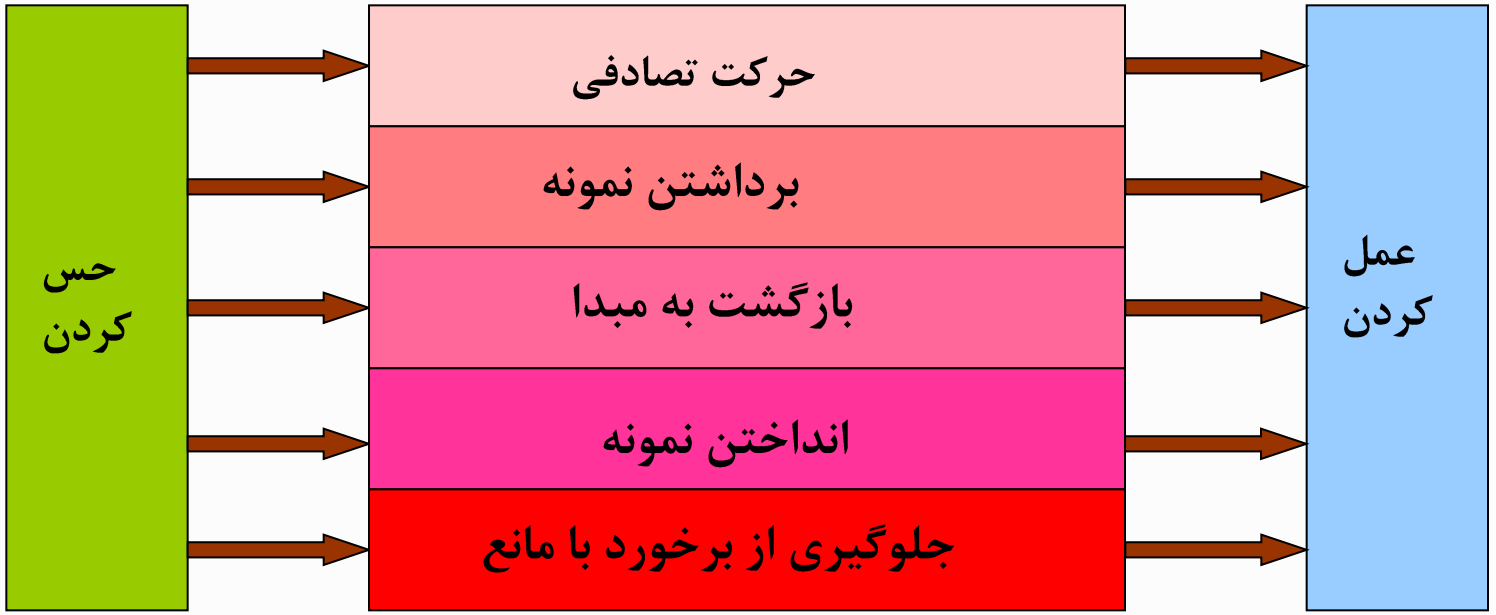
    عمل → وضعیت
  - هر رفتار برای گرفتن کنترل عامل با دیگران رقابت می کند.
  - سطوح بالاتر رفتارهای انتزاعی تری را بروز می دهند
  - لایه های پایین تر قادر به جلوگیری لایه های بالاتر هستند یعنی تقدم بیشتری نسبت یه آنها دارند.

عامل های واکنشی (ادامه)

- یک مثال: سیستم کاوشگر مریخ
  - شامل گروهی از روبات (عامل) های جمع آوری نمونه سنگ ها از سطح مریخ می باشد که در آن مکان نمونه ها از قبل مشخص نیست اما می دانیم که نمونه ها در حوالی یکدیگر قرار دارند.
  - هر روبات تا جایی می تواند از مبدا دور شود که امکان دریافت سیگنال از مبدا اصلی باشد.
  - ارتباط بین عامل ها بصورت غیر مستقیم از طریق مبدا صورت می گیرد.
  - اگر نمونه جمع آوری شده توسط یک عامل در مسیر انتقال به مبدا رها شود عامل دیگر آن را برمی دارد.
  - هر یک از این عامل ها به تنهایی دارای رفتار خاص می باشند و مهم ترین و پایین ترین سطح (با بالاترین اولویت) پرهیز از موانع می باشد.

عامل های واکنشی (ادامه)

- گروه های رفتاری و سلسله مراتب عملیات عامل:



عامل های واکنشی (ادامه)

- مزایا و معایب معماری واکنشی

- مزایا
  - ساده، مقرون به صرفه، کامل و محکم
- معایب
  - همواره باید اطلاعات کافی از محیط برای هر عامل بصورت محلی فراهم باشد.
  - چون تصمیم گیری براساس اطلاعات محلی صورت می گیرد استفاده از این معماری در حالت های غیرمحلی عمومیت ندارد.
  - یادگیری در این معماری با چنان وسعتی همراه خواهد بود که عملا از کارایی آن می کاهد.

استدلال عملی

- استدلال عملی استدلالی است که جهت گیری آن به سمت اعمال است یعنی فرآیند معین کردن این که چه کاری انجام دهیم.
- تعریف Bratman از استدلال عملی
  - استدلال عملی به وزن دادن به ملاحظات مختلف و متضاد به نفع یا برعلیه گزینه های رقیب هم مربوط می شود که در آن ملاحظات مناسب از آنچه که عامل قصد دارد(برایش ارزش دارد/ به آن توجه دارد) و آن چه که عامل باور دارد فراهم می شود.
- استدلال عملی با استدلال نظری متفاوت است. جهت گیری استدلال نظری به سمت باورها است.

استدلال عملی (ادامه)

- استدلال عملی در انسان شامل دو عمل است:
  - بررسی و قیاس : تصمیم گیری در مورد این که به چه اهدافی می خواهیم برسیم
    - خروجی بررسی و قیاس قصد ها می باشند.
  - استدلال عملی: تصمیم گیری در مورد این که چگونه می خواهیم به این اهداف برسیم

بررسی و قیاس

- عامل چگونه بررسی و قیاس می کند؟
  - ابتدا سعی کنید بفهمید چه گزینه هایی پیش روی شماست
  - از بین آنها گزینه هایی را انتخاب کنید و متعهد به انجام آنها شوید.
    - گزینه های انتخاب شده قصد ها خواهند بود.
- تابع بررسی و قیاس را می توان به دو مولفه تقسیم کرد:
  - **تولید گزینه ها**: در این بخش مجموعه ای از گزینه ها تولید می شود. این کار از طریق تابعی به نام *option* انجام می شود که باورهای کنونی عامل و قصد های کنونی آن را می گیرد و مجموعه گزینه ها را تعیین می کند.
  - **فیلتر کننده گزینه ها**: در این بخش تابعی به نام *filter* بین حالت ها و پیشنهادهای مختلف انتخاب می کند و عامل برای رسیدن به آنها متعهد می شود.

استدلال عملی

ایده اصلی آن است که به عامل

- نمایش قصد ها و اهدافی که باید به آنها برسد
- نمایش اعمالی که می تواند انجام دهد و
- نمایش محیط

داده شود تا او برنامه ای را برای رسیدن به هدف تولید کند. در حقیقت این یک برنامه سازی خودکار است.

معماری BDI

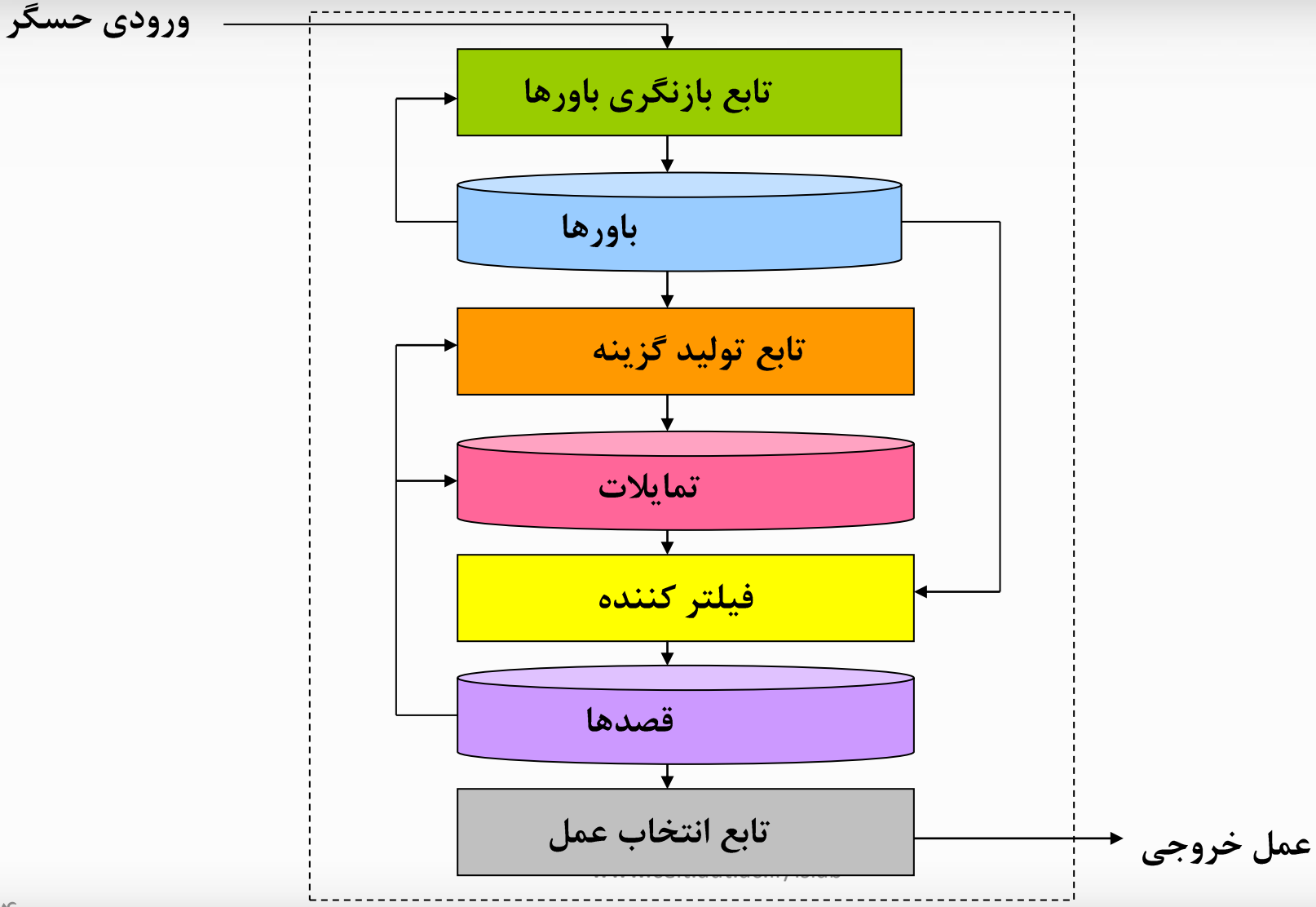
- معماری BDI بر مبنای استدلال عملی است
  - تصمیم گیری در مورد این که به چه اهدافی باید برسیم
  - تصمیم گیری در مورد این که چگونه به آن اهداف برسیم

- BDI
  - باورها (**B**eliefs)
  - خواسته ها (**D**esires)
  - قصد ها (**I**ntentions)

معماری BDI (ادامه)

- قصد ها:
  - از استدلال عملی ناشی می شوند
  - بررسی های آینده را تحمیل می کنند
  - مانا هستند
  - بر روی باورها تاثیر می گذارند که استدلال عملی آینده بر مبنای آنها است

معماری BDI (ادامه)



معماری BDI (ادامه)

- مولفه های عامل BDI
  - **باورها**
    - اطلاعات عامل را در محیط کنونی اش بیان می کند
  - **تابع بازنگری باورها**
    - درک عامل را با نگاشت بر روی محیط دریافت کرده و باورهای جاری را به روز می کند.
    - *brf*: φ(Bel) * P → φ(Bel)
  - **تابع تولید گزینه**
    - ورودی آن باورها و قصد عامل بوده و براساس آنها انتخابهای ممکن برای عامل را که در واقع همان تمایلات اوست تعیین می کند

options: φ(*Bel*) * φ(*Int*) → φ(*Des*)

معماری BDI (ادامه)

- مجموعه ای از **گزینه های معتبر(تمایلات)** که مبین اعمالی است که عامل می تواند انجام دهد.
- یک **تابع فیلترکننده** که ورودی آن باورها و قصدهای عامل بوده و خروجی آن براساس فرآیند تبادل نظر(قیاس) اهداف (قصدهای) جدید عامل است

$filter: \varphi(Bel) * \varphi(Int) * \varphi(Des) \rightarrow \varphi(Int)$

- مجموعه ای از **قصدهای جاری** که کانون فعالیت عامل را تعیین می کند.
- **تابع انتخاب عمل** بر پایه قصد و اهداف فعلی که عملی را که باید انجام شود تعیین می کند

$execute: \varphi(Int) \rightarrow A$ or $action: P \rightarrow A$

معماری BDI (ادامه)

- شبه کد تابع *action* در BDI:

```
function action(p: P) : A
begin
        B := brf (B,p)
        D:= options(B,I)
        I:= filter(B,D,I)
        return  execute(I)
end function action
```

مشکلات BDI

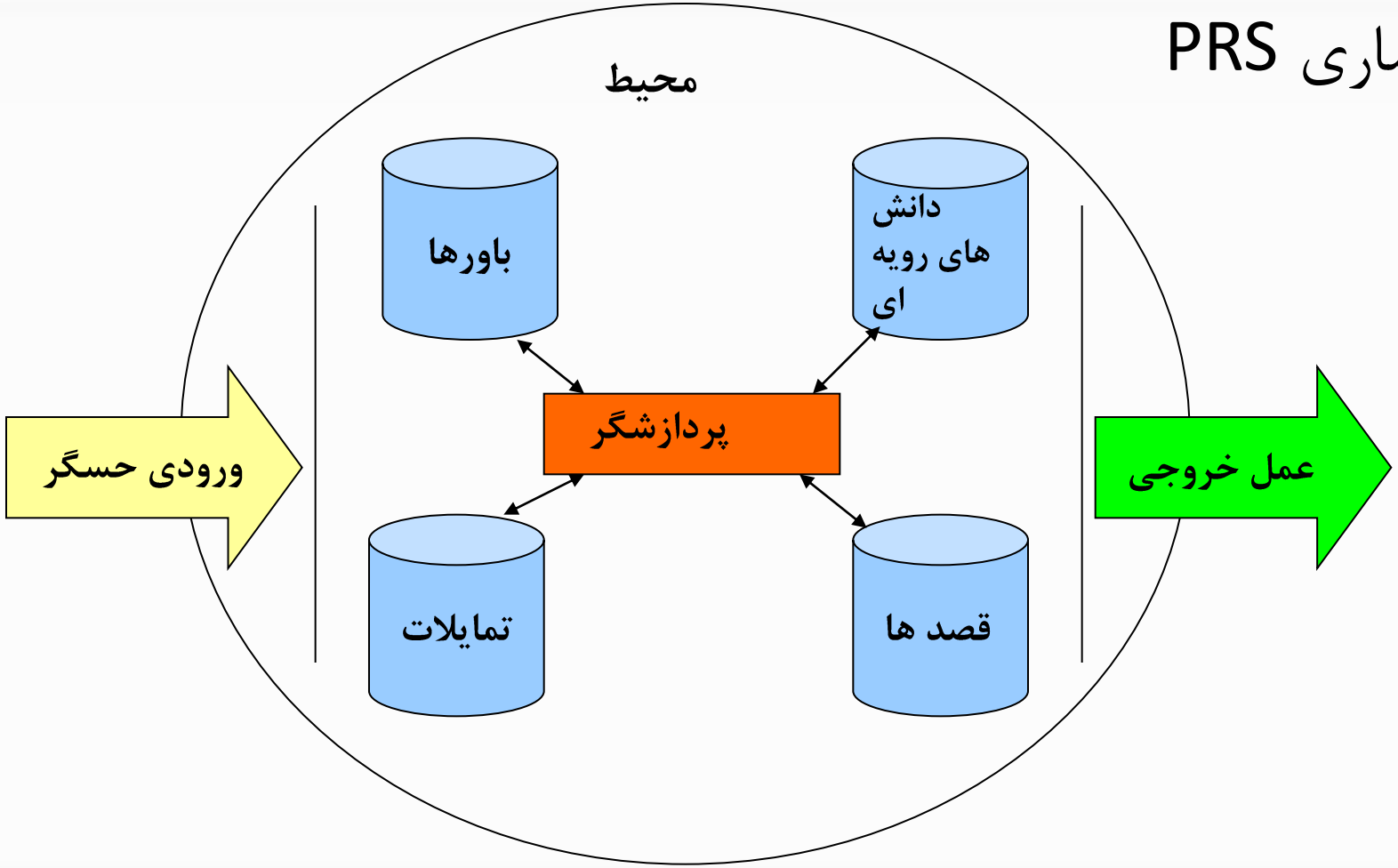
- عاملی که در قصدهای خود تجدید نظر نمی کند تلاش می کند حتی پس از آنکه روشن شد که قابل دستیابی نیستند یا دیگر دلیلی برای رسیدن به آنها وجود ندارد برای رسیدن به آنها تلاش می کند
- عاملی که پیوسته در قصدهای خود تجدید نظر می کند زمان و منابع را به هدر می دهد.

مثالی از یک عامل BDI پیاده سازی شده: عامل PRS

- در این سیستم هر عامل دارای مجموعه ای از برنامه ریزی های انجام شده(plan library) می باشد که بیانگر دانش رویه ای عامل است.
  - دانش رویه ای دانشی درباره مکانیزم هایی است که می توانند توسط عامل به منظور تحقق قصد هایش به کار روند.
- گزینه های پیش روی یک عامل مستقیما توسط برنامه ریزی های آن تعیین می شوند. عاملی که برنامه ریزی ندارد گزینه ای نخواهد داشت.
- در این سیستم عامل ها بازنمایی صریحی از باورها، تمایلات و قصد ها و نیز دانش رویه ای دارند.

عامل PRS (ادامه)

- معماری PRS



مثالی از یک عامل BDI پیاده سازی شده: IRMA

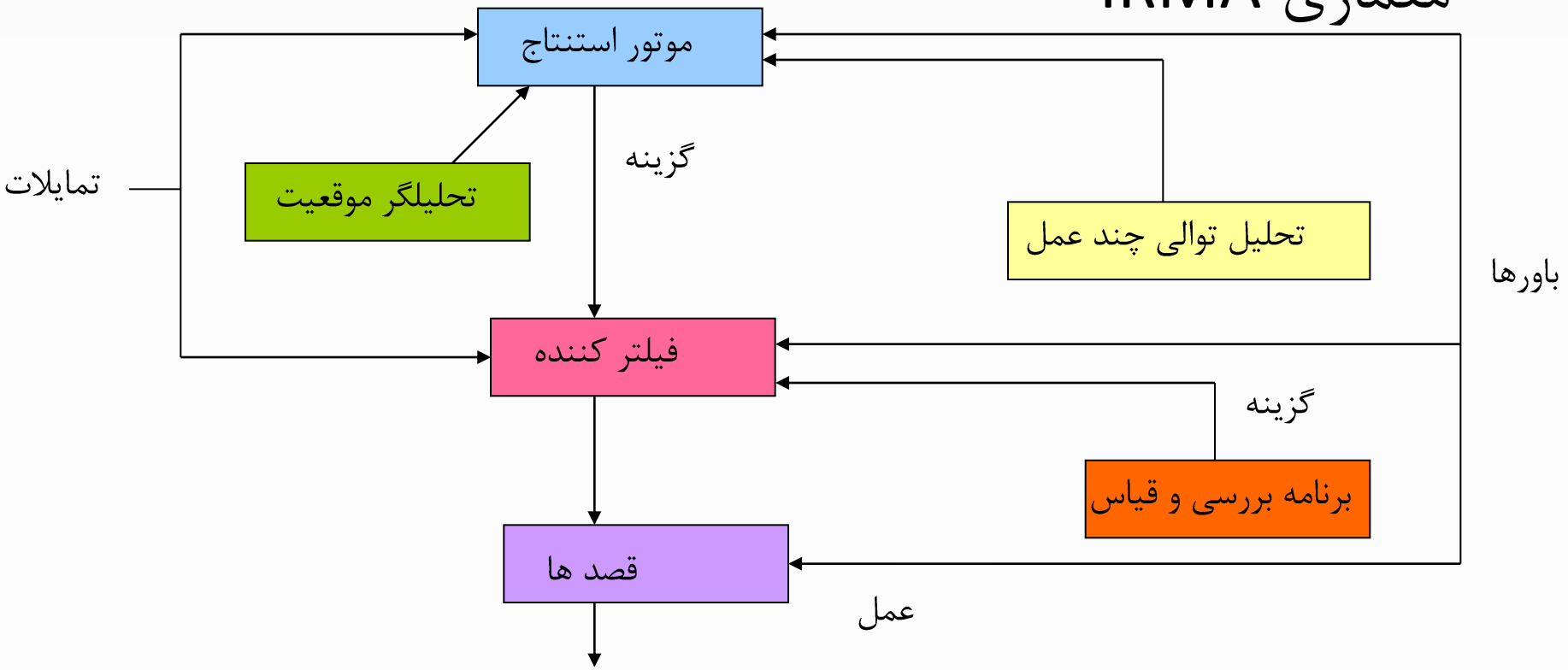
- IRMA چهار ساختمان داده نمادین دارد:
  - مجموعه ای از برنامه ریزی ها
  - نمایش صریح از باورها: اطلاعاتی که عامل در اختیار دارد که یا می تواند بصورت نمادین بیان شود حتی می تواند بسادگی تعریف متغیرهای زبان پاسکال باشد.
  - تمایلات: مفاهیمی که مورد نظر عامل است
  - قصدها: تمام اهدافی که عامل به آنها دسترسی داشته و برای رسیدن به آنها تعهد دارد.

IRMA (ادامه)

- معماری آن قسمت های زیر را دارد:
  - **یک بخش استدلال**: در واقع یک موتور استنتاج است که برای استدلال درباره محیط اطراف عامل بکار می رود.
  - **یک تحلیل گر عملی**: تعیین می کند که کدام برنامه برای رسیدن به قصد انتخاب شود
  - **یک تحلیل گر موقعیت شناس**: نظارت بر محیط در صورت ارائه انتخاب های جدید را دارد
  - **یک برنامه فیلتر کننده**: تعیین می کند کدام گزینه با قصد جاری سازگار است
  - **یک برنامه بررسی و قیاس**: مسئول تصمیم گیری در مورد بهترین قصد برای انجام است.

IRMA (ادامه)

- معماری IRMA



معماری چندلایه (ترکیبی)

- بسیاری از محققین بر این عقیده اند که نه یک روش واکنشی و نه یک روش قیاسی به تنهایی برای ساختن عامل ها کافی نمی باشد.
- انان سیستم های چند لایه ای را پیشنهاد داده اند که رابطه نزدیکی با روش های کلاسیک دارد.
- یک رویکرد بدیهی این است که عامل را از دو یا چند زیر سیستم بسازیم:
  - **یک بخش قیاسی** شامل یک مدل نمادین از جهان که برنامه ریزی ها را ایجاد می کند و از روش ارائه شده در هوش مصنوعی نمادین تصمیم گیری می کند.
  - **یک بخش واکنشی** که بدون استدلات پیچیده قادر به واکنش در برابر رویدادها می باشد.

معماری چندلایه (ادامه)

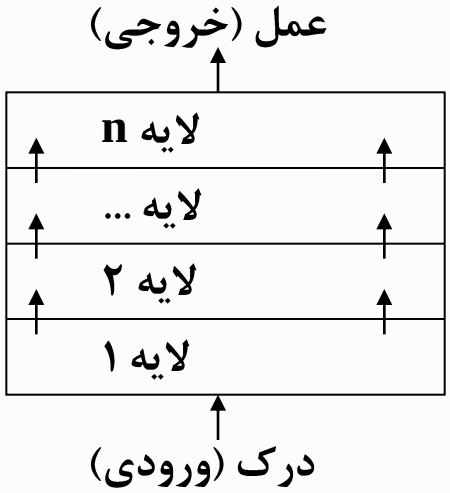
- غالبا به مولفه واکنشی نوعی تقدم نسبت به مولفه قیاسی داده می شود.
- این نوع ساختار بصورت طبیعی به معماری لایه ای منجر می شود که نمونه ای از آن ماشین تورینگ می باشد.
- در چنین معماری ای زیر سیستم کنترلی عامل بصورت یک سلسله مراتب سازماندهی می شود که در آن لایه های بالاتر با اطلاعات انتزاعی تری سروکار دارند.

معماری چندلایه (ادامه)

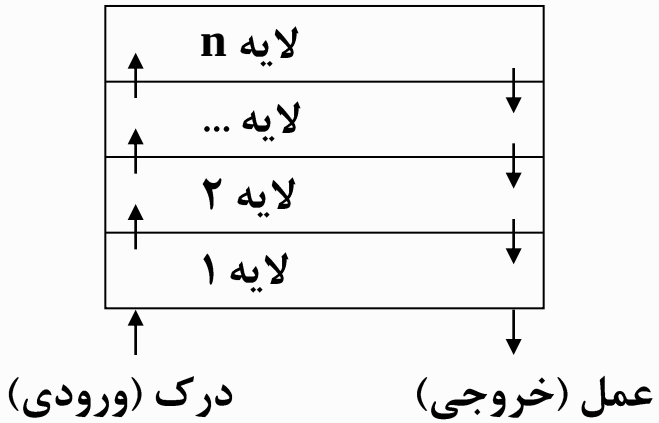
- یک مشکل کلیدی در این گونه معماری ها این است که چه نوع چارچوب کنترلی ای برای مدیریت تعاملات و ارتباطات بین لایه های گوناگون باید در زیرسیستم های عامل تعبیه شود.
- لایه بندی افقی:
  - در این لایه بندی معماری متشکل از دو لایه یا بیشتر است که هر کدام مستقیما به یک حسگر ورودی و یک عملگر خروجی متصل اند.
  - هر لایه به تنهایی همانند یک عامل عمل می کند که پیشنهاداتی راجع به این که چه عملی باید انجام شود تولید می کند.

معماری چندلایه (ادامه)

- لایه بندی عمودی
  - در این لایه بندی ورودی ها و خروجی ها حداکثر با یک لایه مرتبط می باشند. این ساختار به دو نوع تک مسیره و دو مسیره تقسیم می شود.



کنترل تک مسیره



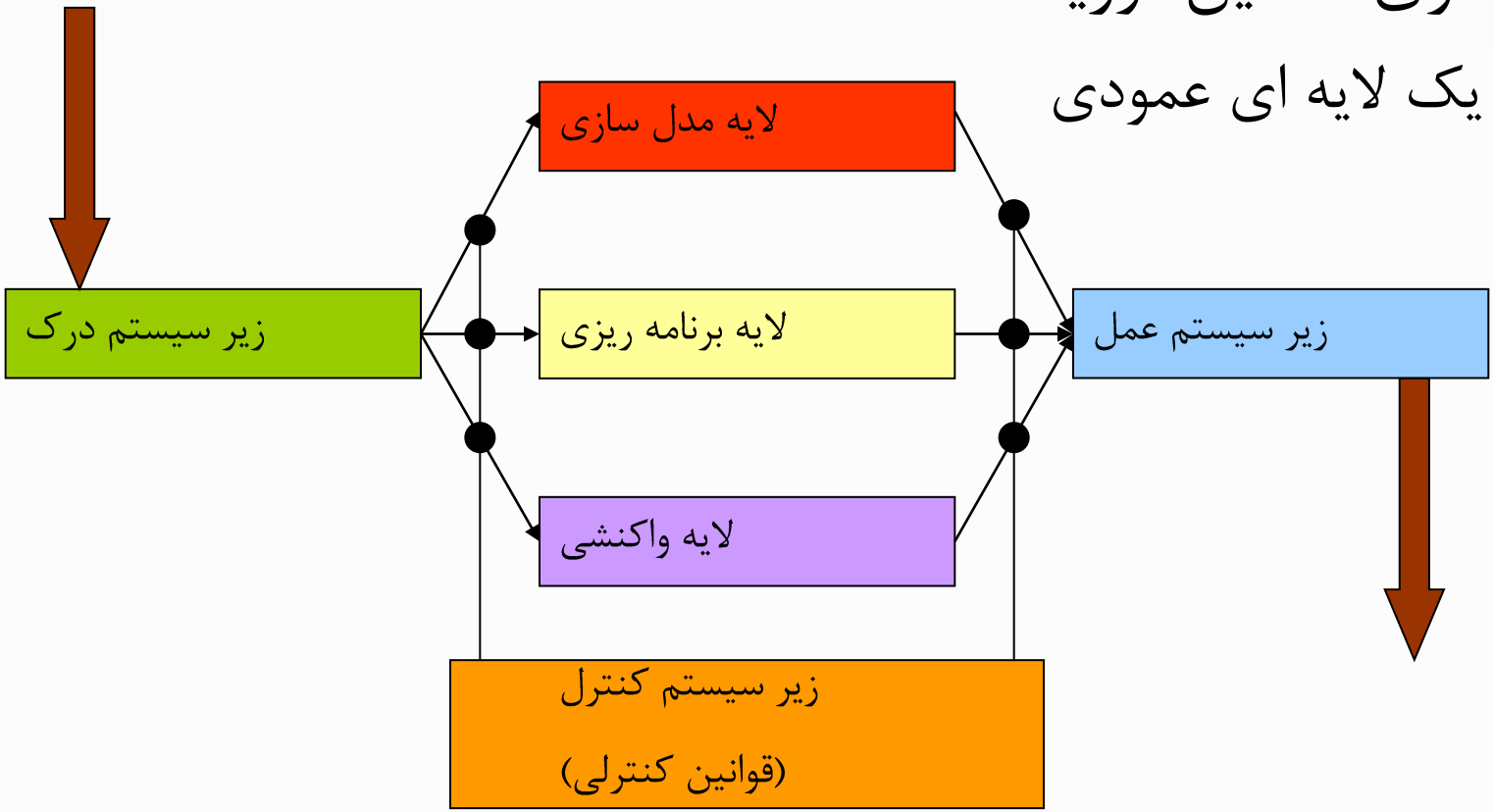
کنترل دو مسیره

معماری چندلایه (ادامه)

- ماشین تورینگ
- شامل سه زیر سیستم درک، عمل و کنترل است که بطور مستقیم با محیط در ارتباط می باشند.
- سه لایه کنترلی دارد که در یک چارچوب کنترلی هماهنگ کننده لایه ها واقع شده اند
  - **لایه واکنش**: نسبت به تغییرات محیط واکنش نشان می دهد. پیاده سازی این لایه بصورت مجموعه ای از قوانین در قالب ساختار جزء به کل می باشد.
  - **لایه برنامه ریزی**: رفتار هدفمند عامل را پیاده سازی می کند و برای این کار از یک کتابخانه از برنامه ها و طرح ها استفاده می کند.
  - **لایه مدل ساز**: شامل بازنمایی نمادین وضعیت های شناختی موجودیت های دیگر در محیط عامل است.
- ارتباط بین لایه ها با استفاده از زیر سیستم کنترلی صورت می گیرد که تعیین می کند کدام لایه کنترل عامل را بدست گیرد.

معماری چندلایه (ادامه)

- معماری ماشین تورینگ
  - یک لایه ای عمودی



منابع

- مقدمه ای بر هوش مصنوعی توزیع شده
  - معرفی عامل و سیستمهای چند عامله
- تالیف
  - دکتر عبدالله زاده بارفروش
  - بهروز معصومی
  - محمدرضا آیت الله زاده شیرازی

با تشکر از توجه شما