Description of the Hidden State of the World

Dimiter Dobrev
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
d@dobrev.com

For an AI to become self-aware, it must answer the questions “Where am I?” and “What's going on?” The answer to these questions is hidden in the internal state of the world. To understand the world is to describe its internal state and the function that determines the transitions from one internal state to another. If an AI doesn't try to understand the world, then it's a weak AI. The way to create strong AI is through describing the internal state of the world.

To create Artificial General Intelligence (AGI) it is not enough to learn to describe the internal state of the world. We still need to move from one-step to multi-step reasoning. This means starting from the current state of the world and mentally taking a few steps forward into the future and thus choosing the best development for us.

Keywords: Artificial General Intelligence, Language for description of worlds, Hidden state of the world, Event-Driven model.

Въведение

Ние се интересуваме от отговора на въпроса „Какво става?“ Тоест ние се интересуваме от текущото състояние на света. Когато гледаме новините по телевизията ние получаваме информация за това какво е текущото състояние на света. Какво се е случило преди 100 години също е интересно, но по-важно е какво се случва в момента.

Състоянието на света може да се раздели на две части – видима и скрита. Имаме текущото наблюдение, което е видимата част. Въпросът е има ли нещо скрито, което не се вижда? Нещата много се опростяват, ако приемем че няма нищо скрито. Този подход е известен под името *Full Observability* и това е най-често използваният подход в ИИ.

В тази статия ще възприемем по-слабо разработения подход на *Partial Observability*, в който се предполага, че състоянието има и нещо скрито. Това е по-интересният подход, защото описанието на скритата част може да го оприличим на фантазия. Когато си представяме нещо невидимо, тогава ние използваме нашето въображение.

Статията се казва „Описание на скритото състояние“, за да подчертаем, че предполагаме съществуването на невидима част. Всъщност ние ще се опитваме да опишем цялото състояние на света без да го делим на видима и скрита част.

Забележка: Понякога вместо „свят“ казват „обкръжаваща среда“ или „природа“.

Нуждаем ли се от памет?

Този въпрос изглежда риторичен. Истинският интелект (човекът) се нуждае памет, следователно и изкуственият човек (AGI) също трябва да има памет.

Въпреки, че отговорът на този въпрос изглежда очевиден, повечето съвременни разработки разглеждат ИИ като устройство без памет (като функция, която зависи само от наблюдението, но не зависи от това, което се е случило до момента).

Това е подходът *Full Observability*, при който се предполага, че няма скрита информация. Тоест всичко важно е видимо в наблюдението. Ако има нещо скрито, то това не е важно и от него не зависи бъдещето развитие.

Имаме ли нужда от памет, когато виждаме всичко важно? Разбира се, че не. Паметта ни е нужна само за неща, които сме видели в миналото, но в момента вече не са ни пред очите.

Когато виждаме всичко, може ли светът да променя вътрешното си състояние. Тоест ще има ли светът памет (вътрешното състояние на света е паметта на света). Отговорът е, че светът ще може да променя вътрешното си състояние, но неговото състояние ще е видимо в наблюдението. Ако се учехме от грешките си (или от собствения си жизнен опит) тогава би било важно какво е новото състояние на света. Тогава бихме разделили състоянията на света на по-добри и по-лоши и бихме се стремили да доведем света до едно по-добро състояние. При подхода *Full Observability* ние предполагаме, че не се учим от собствения си жизнен опит, а от опита на един експерт, който знае точно какво трябва да се направи. Затова ние няма да се опитваме да доведем света до по-добро състояние, а само ще се стремим да имитираме действията на експерта, защото той знае точно какво трябва да се направи.

Тоест при *Full Observability* ние няма да се интересуваме от вътрешното състояние на света или за по-просто ще предполагаме, че светът има само едно вътрешно състояние (тоест, че няма памет).

При *Full Observability* ние се опитаме да имитираме поведението на експерта на базата на неговия жизнен опит. Жизненият опит на експерта ще има вида на множество от наредени двойки $\langle observation, action \rangle$. Защо множество, а не списък? Защото редът на двойките няма значение.

Жизненият опит на експерта има вида:

$$\{ \langle o_i, a_i \rangle \mid i \in I \}$$

Нека поведението на експерта се описва с една функция f.

$$\forall i \in I \quad f(o_i) = a_i$$

Ще търсим функцията f', която е апроксимацията на f. Тоест ще търсим най-простата функция, за която:

$$\forall i \in I \quad f'(o_i) = a_i$$

Тази апроксимация обикновено търсим в множеството на невроните мрежи. Този метод ни дава удивителни резултати. Получаваме много умни програми, но все пак това са програми без памет (функции), а както казахме AGI трябва да има памет.

Partial Observability

Имаме случаи, когато паметта на ИИ е задължителна. Нека имаме жизнен опит, който е последователност от действия и наблюдения:

$$o_0, \quad a_0, \quad o_1, \quad a_1, \quad \ldots \quad , o_{t-1}, \quad a_{t-1}, \quad o_t$$

Нека светът променя вътрешното си състояние и нека част от вътрешното му състояние е скрита и не се вижда в поредното наблюдение. Нека предположим, че скритата част е важна (тоест, че от нея зависи бъдещото развитие). Тогава ИИ ще трябва да помни какво се е случило до момента. Тоест ще трябва да предположим, че ИИ е устройство с памет (или функция с памет). Тоест функция, която от текущата стойност на паметта си и от текущото наблюдение ще върне следващото действие и новата стойност на паметта си.

$$f(m_i, o_i) = \langle a_i, m_{i+1} \rangle$$

Към редицата от действия и наблюдения ще добавим стойностите на паметта на ИИ, както и вътрешните състояния на света:

$$m_0, o_0, s_0, a_0, m_1, o_1, s_1, a_1, \quad \ldots \quad , o_{t-1}, s_{t-1}, a_{t-1}, m_t, o_t, s_t$$

Светът ще се описва от началното си състояние s_0 и една функция, която от текущото състояние на света и поредното действие на ИИ ще върне следващото наблюдение и новото състояние на света.

$$g(s_i, a_i) = \langle o_{i+1}, s_{i+1} \rangle$$

Забележка: Тук ще отбележим, че s_0 може да зависи от o_0. (Това е така, ако състоянието „помни“ последното наблюдение.) Затова ще кажем, че светът се описва от o_0, s_0 и g.

При *Full Observability* ние се опитвахме да апроксимираме функцията f, а сега ще апроксимираме g. Тоест вместо да се опитваме да опишем експерта (с чийто жизнен опит разполагаме), ние ще се опитаме да разберем (да опишем) света. Сега ние не можем да апроксимираме експерта, защото жизненият опит, с който разполагаме, може да не е жизненият опит на някакъв експерт, а да е произволен жизнен опит. Може това да е нашият собствен жизнен опит, в който има много грешки. Този жизнен опит няма да е пример за подражание.

Да разберем света означава да намерим функцията g и текущото състояние s_t. Това ние не можем да го намерим точно и затова ще го намерим приблизително. Тоест ще намерим функцията g' и текущото състояние s'_t, които са техните апроксимации.

Паметта на ИИ ще бъдат g' и s'_t. Тоест ние ще забравим за паметта на ИИ и ще изследваме редицата, в която паметта на ИИ не участва:

$$o_0, s_0, a_0, \quad o_1, s_1, a_1, \quad \ldots \quad , o_{t-1}, s_{t-1}, a_{t-1}, \quad o_t, s_t$$

Ако успеем да направим програма, която разбира света (намира g' и s'_t) ние ще накараме тази програма да изследва бъдещето (да направи няколко стъпки напред) и да избере действието, което води до най-доброто развитие на света (кое е най-доброто развитие е друг въпрос). Това ще бъде търсеният от нас ИИ.

Как ще опишем текущото състояние

Нека S да бъде множеството на вътрешните състояния на свата. Това е едно множество, за което не знаем нищо.

Нека S' да бъде множеството на въображаемите състояния на свата. Това е едно конкретно множество, което сме избрали. Чрез това множество ние ще апроксимираме множеството S. Множеството S' може да се променя във времето, но за всеки конкретен момент t ние ще сме избрали конкретно множество и това ще е нашата представа за възможните състояния на света в този момент.

Достижимите състояния на света са изброимо много (тръгваме от е s_0). Затова можем да приемем, че множеството S' е изброимо. Всяко кодиране на S' е възможно и затова можем да приемем, че $S' = \mathbb{N}$. Дори бихме могли да приемем:

$$s'_i = i$$

Това е едно възможно решение, но идеята не е добра. Първо това кодиране зависи от t и на следващата стъпка ще трябва да го променим. По-сериозният проблем е, че по този начин опростяваме максимално представянето на вътрешното състояние на света и пренасяме цялата сложност върху функцията g'. Всяка пермутация (кодиране) на вътрешните състояния на света е възможна, но кодирането може да доведе до усложняване. По-добре би било вътрешното състояние да има по-сложна структура и за сметка на това функцията g' да е по-проста.

Как изглежда вътрешното състояние на една програма написана на C++ или на Java? Това състояние се описва с текущите стойности на променливите. Тест вътрешното състояние може да се представи като n-торка от скалари. Тоест можем да приемем, че $S' = \mathbb{N}^n$. Нека да добавим, че програмите освен променливи имат и масиви. Затова част от скаларите ще заменим с k-торки. По този начин получаваме едно изброимо множество S', което ще опише вътрешното състояние на конкретна програма (на нейните променливи и масиви).

Състоянията на програмите са изброимо много, защото те използват крайна памет (дори машината на Тюринг в конкретен момент използва само крайна част от лентата си). Ако искаме да опишем свят, който използва безкрайна памет ще трябва да предположим, че множеството S' е неизброимо. Представете си машина на Тюринг, която започва работа си от лента, която цялата е пълна с информация (а не само крайна част от лентата, както е обикновено). За да опишем вътрешното състояние на такава машина ще трябва да си представим неизброимо много вътрешни състояния, които съответстват на състоянията на безкрайната лента, които са континуум много.

Затова ще се откажем от предположението, че множеството на въображаемите състояния S' е изброимо и ще приемем, че то описва стойността на променливи и на масиви, като масивите могат да бъдат както крайни така и безкрайни.

Тогава какво ще представлява състоянието s'_t? Това няма да е конкретен елемент на S'. Вместо това s'_t ще бъде описание на множество от състояния на S'. Например, ако имаме безкрайна лента пълна със символи, то описание на състоянието на тази лента може да бъде конкретна стойност за първите 10 символа и „неизвестна“ стойност за останалите символи.

Големите езикови модели (LLM)

Когато имаме разговор и трябва да отговорим на поредния въпрос, на нас ни е нужно да вземем предвид целия разговор, а не само последния въпрос. Тоест в този случай не можем да търсим ИИ като функция, която на поредния въпрос дава поредния отговор. Все пак, дори и в този случай успяваме да сведем задачата до *Full Observability*. За целта просто приемаме, че наблюдението не е последния въпрос, а е целият разговор до момента. Жизненият опит на експерта изглежда като множество от наредени двойки състоящи се от разговор до момента и продължение на разговора $\langle conversation, continuation \rangle$.

Разбира се, в този случай, ако искаме да търсим ИИ като апроксимираща функция, ще трябва да имаме огромен брой разговори и техните продължения, при това тези продължения трябва да са правилните продължения, т.е. да са направени от експерт, който казва това, което трябва. Щом ще имитираме експерта, трябва да сме сигурни, че разполагаме с жизнения опит на някой, който действително е експерт.

Някои казват, че LLM не е функция, защото на едно начало на разговора не дава едно възможно продължение, а дава безброй възможни продължения. Истината е, че това е функция, която връща вероятностно разпределение (таблица с вероятностите на различните продължения). След това случайно се избира едно продължение (според разпределението). Все едно да имате функция, която да връща таблица с k реда и на реда i да стои вероятността p_i. След това чрез монета да определяте j (с вероятност p_j). Тази функция в комбинация със случайността вече не е функция, но съществената част е функция.

Пример за LLM

Нека да вземем пример за LLM като направим програма, която играе шах без дори да разбира каква е позицията на фигурите върху таблото. За да опишем света трябва да кажем кой е агентът, който живее вътре в този свят и мести фигурите срещу нас. Нека нашият противник е програма, която изчислява три хода напред и избира двата най-добри хода. След това хвърля монета и решава кой от двата хода да играе. (Бихме могли да предположим, че противникът винаги играе най-добрия си ход, но тогава светът би бил детерминистичен, а това би било твърде елементарно.)

Нека „ход“ да бъде наредена четворка $\langle x_1, y_1, x_2, y_2 \rangle$, в която $\langle x_1, y_1 \rangle$ са координатите на позицията от която вдигаме фигура, а $\langle x_2, y_2 \rangle$ са координатите на позицията, на която поставяме вдигнатата фигура. За всяка координата имаме 8 възможности. Следователно възможните ходове са $2^{12} = 4096$, но само малка част от възможните ходове са коректни.

Нека експертът, чийто жизнен опит ще имитираме, да бъде програмата, която изчислява десет хода напред и избира най-добрия ход. Нека жизненият опит се състои от голям брой партии между експерта и противника (всяка партия, заедно с всичките и начала завършващи с ход на експерта). В двойките действието ще бъде ходът на експерта, а наблюдението ще бъде партията от началото до този ход.

Тук резултатът от апроксимацията няма да е функцията f, а ще бъде:

$$f^*(\langle o_0, a_0, \ldots, o_{i-1}, a_{i-1}, o_i \rangle) = a_i$$

Функцията f^* по естествен начин се определя от f и от m_0. Тази функция има само един аргумент, макар че този аргумент е списък състоящ се от много ходове.

За да получим функция f^*, която играе сравнително добре трябва да имаме огромен жизнен опит (огромен брой партии изиграни между експерта и противника). Много трудно ще е дори да направим програма, която играе само коректни ходове. За да играе коректни ходове програмата трябва да разбере позицията на таблото, а това никак няма да е лесно.

Това е проблем на подхода *Full Observability*. При този подход, за да „се научим“ ни е нужен огромен жизнен опит. Тоест програмата се учи много бавно (бавно не в смисъл на време, а на количество информация или на брой стъпки в жизнения опит). При човека това не е така. Може би това е така на ниско ниво, когато човека възприема визуална и звукова информация, но на високо ниво нещата са различни. Достатъчни са само няколко примера, за да успее човекът да схване някоя зависимост. Често е достатъчно на човека да кажете нещо само един единствен път и той вече ще го знае и ще може да го използва.

Ако успеем да намерим функция f^*, която играе само коректни ходове, то тя вероятно ще играе доста силни ходове, защото тази функция ще имитира експерта, а той играе добре. Тоест големият проблем е да разберем състоянието на света (позицията на таблото), а силната игра ще я получим като страничен ефект.

Въпросът на Антон

Моят колега Антон Зиновиев ми зададе следния въпрос: „Можем ли с апроксимация вместо функцията f^* да търсим функцията g^*, която описва света?“

$$g^*(\langle o_0, a_0, \ldots, o_{i-1}, a_{i-1} \rangle) = o_i$$

Функцията g^* по естествен начин се определя от g и от s_0. Разбира се, g^* ще е многозначна функция, защото противникът избира един от двата най-добри хода. Това не е проблем, защото по принцип приемаме, че при LLM продължението е многозначна функция.

Ако имаме g^*, чрез нея ще можем да получим функцията f^* (като направим няколко стъпки напред в бъдещето). Тоест въпросът е дали можем чрез апроксимация да търсим обяснение на света. Отговорът зависи от това с какъв жизнен опит разполагаме. Ще разгледаме два случая:

1. Ако имаме жизнения опит на някой експерт. В този случай защо да апроксимираме g^* и по този начин индиректно да търсим f^*. По-естествено би било да апроксимираме f^* и по този начин да я намерим директно.

Забележка: Чрез жизнения опит на експерта трудно ще намерим g^*, защото този жизнен опит е твърде беден (във всяка позиция имаме само един възможен ход на експерта). Тази информация може да не е достатъчна за описанието на g^*, въпреки че с аналогия може да се опитаме да познаем какво ще се случи, ако играем ход различен от този на експерта (апроксимацията ще ни даде такава аналогия).

2. Нека разполагаме с жизнения опит на случайния играч. Също така можем да предположим, че имаме жизнения опит на необучения ИИ, който се лута, докато опознава света и прави много грешки докато е необучен.

В този случай ще трябва към входа да добавим още четири наблюдения: „победа“, „загуба“, „реми“ и „некоректен ход“. На наблюденията ще сложим оценки (награди). Победата ще има положителна оценка, а на загубата оценката ще бъде отрицателна. Най-ниската оценка ще е на „некоректен ход“, защото трябва да предпочетем загубата пред това да играем некоректен ход. Дали оценката на „реми“ ще е положителна или отрицателна ще зависи от това дали ще се стремим към реми или ще бягаме от него. Останалите 4096 наблюдения (ходовете на противника) ще имат оценка нула.

Когато имитирахме експерта, тези допълнителни наблюдения не ни бяха нужни. Експертът и светът не могат да играят некоректни ходове, а когато свършваше партията нямаше нужда да знаем как тя е свършила, защото ние просто имитирахме експерта, а той знаме какво да прави. Сега обаче функцията g^* ще трябва да предсказва тези наблюдения, за да можем да изберем най-доброто развитие на света.

Партията може да свърши и след ход на противника. В този случай може да поискаме ИИ да признае загубата (или ремито) и това да е единствения му коректен ход. За да не добавяме още действия, ще приемем че в този случай всички ходове на ИИ са коректни и следващото наблюдение ще е „загуба“ (или „реми“ съответно).

По този начин чрез тези четири допълнителни наблюдения може да направим жизнен опит, чрез който може да се апроксимира функцията g^*.

Разделяне на живота на игри

В горния пример много важно е това, че разделихме живота (жизнения опит) на отделни игри (партии), които започват от една и съща начална позиция на таблото. При това разделяне последователността, в която се играят игрите няма значение. По този начин можем да представим жизнения опит като едно дърво с много разклонения.

Ако жизненият опит е една последователност от наблюдения и действия, която не може да се раздели на отделни игри, тогава жизненият опит ще се представи като един път без разклонения (или дърво без разклонения).

Ако искаме да търсим f^* и g^* чрез апроксимация е много важно да разделим живота на отделни игри. Аргументът на тези функции е текущата игра и ако имаме само една игра, то тогава аргументът ще бъде целият жизнен опит, а тази информация е твърде много. По-лесно се търси апроксимация, когато жизненият опит е едно добре разклонено дърво, защото тогава имаме повече информация за това как се държи светът (и експертът).

Можем ли да разделим жизнения опит на отделни игри? За целта трябва да намерим моменти, в които състоянието на света е едно и също. По принцип чрез събитийните (ED) модели ние правим класификация на състоянията и ги разделяме на групи от подобни състояния. Разбира се, това, че две състояния са подобни по някакъв признак въобще не означава, че те са еднакви. Спокойно можем да предположим, че всички състояния в живота са различни. Както е казал Хераклит „Не можеш да влезеш два пъти в една и съща река“.

Друго възможно предположение е, че животът се състои от отделни игри, но те не започват от едно и също състояние. Тогава жизненият опит изглежда като отделни нишки, които не са свързани. При това предположение разбирането на света ще е още по-рудно. В началото на всяка нишка ще трябва първо да се ориентираме и да разберем кое е текущото състояние на света.

Разказаха ми историята за един пияница, които живее два паралелни живота. Единият докато е трезвен, а другият докато е пиян. Той помни само това, което се е случило докато е бил трезвен и затова живота му се състои от отделни нишки, които не са свързани помежду си. Всеки път, когато се „осъзнае“ (когато започне нова нишка) той трябва да установи какво е текущото състояние на света (къде е, какво е здравословното и финансовото му състояние и т.н.). Разбира се, текущото състояние на света се променя скокообразно, но функцията g остава постоянна (например земното притегляне се запазва). Не е съвсем ясно каква част от света е описана в s_t и каква част е във функцията g. Например, ако пияницата се „осъзнае“ в космоса може и земното притегляне да го няма.

Компресия

По принцип можем да приемем, че функцията g^*, е търсената от нас функция g, като състоянието на света е текущата игра до момента (или целият жизнен опит, ако не можем да разделим живота на игри).

Текущата игра носи цялата информация, която ни е нужна, за да определим състоянието на света, но тази информация е твърде голяма и необработена. Тази информация трябва да се компресира и силно да се намали. Става дума за компресия със загуба на информация, защото трябва да се извлече същественото и да се изхвърли маловажното (не можем да помним всичко).

Кое е съществено и кое е маловажно? Това ще се определи от модела на света, които ще намерим.

Функцията g

Казахме, че може да си мислим, че множеството на въобразаемите състояния S' е изброимо. Множествата на действията (Σ) и на наблюденията (Ω) са крайни (или изброими). Следователно g', която апроксимира g е функция между две изброими множества.

$$g' : S' \times \Sigma \rightarrow S' \times \Omega$$

С някакво кодиране можем да си мислим, че g' е функция от естествените числа в естествените числа.

$$g' : \mathbb{N} \rightarrow \mathbb{N}$$

Проблемът е, че броят на тези функции е континуум много. Ние ще търсим описание на функцията g' и това описание трябва да го търсим в някакво изброимо множество на описуеми функции. Най-подходящият кандидат за такова множество са изчислимите функции. Този подход е възприет в [7, 6, 2], където се предполага, че функцията на света се апроксимира с изчислима функция.

Въпреки всичко да търсим g' в множеството на изчислимите функции не е добра идея, защото това предполага, че светът е детерминиран. Много по-добре е да предположим, че в света има случайност и че функцията g' може да вика един оракул, който да дава случайна стойност с някакво разпределение (например да хвърля зар). Добавянето на такъв оракул ни извежда от множеството на изчислимите функции, но получаваме изброимо (описуемо) множество, което е удобно за предсказване на бъдещето. Разбира се, предсказанието е с варианти и различните варианти си имат различна вероятност.

Има още една причина, поради която трябва да излезем от класа на изчислимите функции. Светът може да е многоагентен и функцията g' да зависи от действията на другите агенти. Тези агенти ще представим с оракули, които описват действията им. Тези оракули са някакви неизвестни функции и тяхното добавяне силно опростява описанието на света. Представете си, колко по-просто би било, ако в горния пример противникът беше просто една черна кутия (оракул, който се опитва да играе срещу нас). Вместо това ние описахме една конкретна програма със случайност. Да добавим и това, че когато имаме конкретен свят с конкретен противник е много малко вероятно ние да успеем да го опишем като конкретна програма със случайност. Затова е много по-лесно и по разумно да си го представим като черна кутия.

Класът на програмите с агенти е изброим (описуем) и има хубавото свойство, че с него можем да предсказваме бъдещето. Пак имаме различни варианти за действията на агентите, но може да предположим за някои, че се опитват да ни помогнат, а за други, че се опитват да ни попречат. Тоест предсказанието на бъдещето ще прилича на алгоритъма Min-Max, където ще изчисляваме максимум по възможните действия на агентите приятели и минимум по действията на враговете.

Събитийни модели

Казахме, че ще търсим g' в множеството на изчислимите функции с оракули (които връщат случайност и които емулират агенти). Тоест, ще представим g' като машина на Тюринг или като програма на някой език за програмиране.

Всеки програмист знае, че програмите са нещо твърде крехко и ако променим един ред или дори само един символ от програмата, тя обикновено спира да работи. Програмата прилича на сложен часовников механизъм, в който има много зъбни колела и ако променим размера само на едно от тези колела, часовникът ще спре да работи.

Бихме искали да конструираме g' от такива части, които лесно могат да се комбинират и да се променят. Подходящите части, от които можем да изградим g', се казват събитийни (ED) модели и са описани в [1, 3]. ED моделите са нещо като крайни автомати. Например при машините на Тюринг главата е ED модел.

На базата на ED моделите ще построим въображаемото състояние на света. На всеки ED модел ще съответства един скалар и това ще е текущото състояние на ED модела. Освен това на всеки ED модел ще съответства и един масив (краен или безкраен в зависимост от това, колко са състоянията на ED модела). Състоянието на модела ще е индексът на масива, а стойностите ще се наблюдават, когато ED моделът е в съответното текущо състояние.

Алгоритми

Алгоритмите също ще бъдат ED модели. Това е описано в [5]. Там чрез алгоритми са дефинирани правилата, по които се движат шахматните фигури.

В [8] Yann LeCun казва, че никой не знае как да направи йерархичен алгоритъм. Всъщност, проблемът е да се направи алгоритъм, а след това не е проблем да се организира йерархия между алгоритмите. Как се прави алгоритъм вече е описано в [5].

Алгоритмите са важни за планирането на бъдещето. Казахме, че мислено ще направим няколко стъпки напред, за да изберем най-доброто развитие на света. Въпросът е, че няма да правим малки, а големи стъпки. Малка стъпка е едно действие и прехода от t към $t+1$. Голяма стъпка е изпълнението на един алгоритъм, което обикновено продължава много малки стъпки.

Причината, поради която предпочитаме големите стъпки е, че по този начин ще можем да погледнем много по-далеч в бъдещето. За това планиране ще ни трябва обобщение на g'. Обобщената функция g' ще взима като аргументи представа за състоянието на света и алгоритъм, а като резултат ще върне представата за състоянието на света след изпълнението на алгоритъма.

Алгоритмите могат да се изпълняват и паралелно. Обикновено ние изпълняваме паралелно голям брой различни алгоритми. Например може да чакаме автобуса и паралелно с това да четем вестник. Паралелно с това може и да следваме и в университета (това също е алгоритъм състоящ се от посещение на лекции и взимане на изпити).

Докато чакаме автобуса ние също така можем да планираме деня си. Планирането също е алгоритъм. Повечето неща в ИИ се случват децентрализирано или паралелно. Например търсенето на зависимости, планирането и изпълнението на алгоритми могат да се случват несъзнателно или на ниво подсъзнание. Има едно нещо, което трябва да бъде централизирано и да се извърши съзнателно. Това е изборът на един от плановете или стартирането на един от няколко възможни алгоритъма. (Когато имаме избор. Ако няма избор, няма нужда да му мислим.) Например в главата ни се появяват два плана. Единият е да отидем в университета, а другият е да се разходим в парка. Съзнателно (централизирано) трябва да изберем един от тези два плана и да задействаме изпълнението на съответния алгоритъм. В противен случай ще застанем като Боридановото магаре между университета и парка и няма да можем да отидем никъде.

Заключение

За да опишем състоянието s'_t и функцията g' ни е нужен език за описание на светове. Такъв език има описан в [4, 5]. Този език е на базата на ED моделите, като над тях са надградени абстракции като агент и обект.

Можем ли чрез LLM да създадем AGI? В [8] LeCun казва, че това няма как да стане, защото всеки LLM изчислява крайна функция. (По-точно той казва, че времето за намиране на отговора е константа и не зависи от въпроса, но това е същото, казано по друг начин.)

Напълно сме съгласни с LeCun, че не може да създадем AGI чрез LLM, но не защото LLM е крайна функция, а защото е функция без памет и защото не се учим от собствения си жизнен отпит, а от някакъв експерт, на който се опитваме да подражаваме.

Щом се съгласихме да изградим g' от ED модели, следователно сме се отказали да я търсим в множеството на изчислимите функции и ще се задоволим само с крайните функции. Да търсим g' сред изчислимите функции всъщност не е много добра идея. В този случай бихме получили ИИ който страда от припадъци (забива и за известно време ни приема ни предава). Затова ще приемем, че времето, за което се изчислява g' е ограничено от някакъв максимум. Изчисляването на g' означава от дадена представа за състоянието на света и дадено действие на агента да се получи новото наблюдение и новата представа за състоянието на света (това е при дадени стойности на оракулите).

Функцията g' макар и крайна може да изчисли всичко (произволна програма). Само трябва да дадем на ИИ една безкрайна лента и възможността да чете и да пише върху нея. По този начин ще вградим в g' една универсална машина на Тюринг. По същия начин главата на машината на Тюринг може да изпълни произволна програма, но не я изпълнява за една стъпка, а за много стъпки. Идеята да дадем на ИИ една безкрайна лента съответства на идеята на Тюринг, че неговата машина е човек, на който сме му предоставили безкрайно количество хартия.

Тоест за една стъпка имаме крайна функция, но за много стъпки получаваме изчислима функция. Все пак, иска ни се функцията g' да е нещо повече от крайна функция, но да не забива. Затова ще предполагаме, че имаме оракули, които изчисляват програми. Вече добавихме оракули, които хвърлят зар и които емулират агенти. Нищо не пречи да добавим такива, които изчисляват програми, но, за да не забива g', ще предположим, че тези оракули работят асинхронно. Това означава, че, ако успеят да върнат отговор на същата стъпка, тогава добре, но, ако не успеят, може да върнат отговор на стъпка $t+1$ или на някоя по-късна стъпка, а дотогава отговорът да бъде „неизвестен“.

Ние математиците знаем, че когато доказваме теорема, понякога това отнема години, а понякога въобще не успяваме да докажем. Доказателството на теореми по-скоро прилича на многостъпковото изпълнение на алгоритми, а асинхронното изпълнение на програми, по-скоро прилича на въпрос, чийто отговор не можем да дадем веднага, но след малко отговорът сам се появява в подсъзнанието ни.

References

[1] Dobrev, D. (2018). Event-Driven Models. *International Journal "Information Models and Analyses", Volume 8, Number 1, 2019, pp. 23-58.*

[2] Dobrev, D. (2019). The IQ of Artificial Intelligence. *Serdica Journal of Computing, Vol. 13, Number 1-2, 2019, pp.41-70.*
https://serdica-comp.math.bas.bg/index.php/serdicajcomputing/article/view/sjc.2019.13.41-70/pdf

[3] Dobrev, D. (2021). Before We Can Find a Model, We Must Forget about Perfection. *Serdica Journal of Computing, Vol. 15, Number 2, 2021, pp. 85-128.*

[4] Dobrev, D. (2022). Language for Description of Worlds. Part 1: Theoretical Foundation. *Serdica Journal of Computing 16(2), 2022, pp. 101-150*.
https://serdica-comp.math.bas.bg/index.php/serdicajcomputing/article/view/sjc.2022.16.101-150/pdf

[5] Dobrev, D. (2023). Language for Description of Worlds. Part 2: The Sample World. *Serdica Journal of Computing 17(1), 2023, pp. 17-54*.
https://serdica-comp.math.bas.bg/index.php/serdicajcomputing/article/view/sjc.2023.17.17-54/pdf

[6] Hernández-Orallo, J., & Minaya-Collado, N. (1998). A formal definition of intelligence based on an intensional variant of Kolmogorov complexity. *Proc. intl symposium of engineering of intelligent systems (EIS'98), February 1998, La Laguna, Spain (pp. 146–163). : ICSC Press.*

[7] Hutter, M. (2000). A Theory of Universal Artificial Intelligence based on Algorithmic Complexity. *arXiv:cs.AI/0004001 [cs.AI]* https://arxiv.org/abs/cs/0004001

[8] LeCun, Yann (2024). Lex Fridman Podcast #416.
https://youtu.be/5t1vTLU7s40?si=5VIlr6no8lzBMTM0

[9] Zinoviev, Anton. https://www.fmi.uni-sofia.bg/en/faculty/anton-kirilov-zinoviev