Aktywacje, obliczenia subsymboliczne

Konrad Miazga

Poznan University of Technology

Poznań, 2018

Czy każda wiedza jest sobie równa?

Czy każda wiedza jest sobie równa?

wiedza = chunki

Obliczenia subsymboliczne

Aby poprawnie modelować niepewności związane z pracą mózgu, musimy odblokować dodatkowe mechanizmy ACT-R'a. Aby to uczynić, należy ustawić odpowiednią wartość parametru *:esc*:

```
(sgp :esc t)
```

Aktywacja chunku

Od teraz, z każdym chunkiem i w pamięci deklaratywnej, związany bedzie pewien poziom aktywacji A_i. W razie konfliktów, wybierany będzie zawsze chunk o najwyższej wartości aktywacji. Ponadto, możemy określić próg aktywacji τ definiowany parametrem *:rt* (**r**etrieval **t**hreshold) który będzie oznaczał minimalny poziom aktywacji chunka wymagany do tego, aby chunk został zwrócony do bufora:

```
(sgp :rt -0.5)
```

Aktywacja chunku

Aktywację chunka liczymy ze wzoru:

$$A_i = B_i + \varepsilon,$$

gdzie:

- B_i to bazowy poziom aktywacji,
- ε to wartość szumu.

Powyższy wzór jest uproszczony – nie bierze pod uwagę kontekstu – ale na chwilę obecną nam wystarczy.

Bazowy poziom aktywacji

Bazowy poziom aktywacji chunka liczymy ze wzoru:

$$B_i = ln(\sum_{j=1}^{n} t_j^{-d}),$$

gdzie:

- n to liczba „prezentacji” chunka i,
- t_j to czas od j-tej prezentacji,
- d to parametr zaniku. Domyślnie wynosi 0.5, można go zmienić (*:bll*).

Prezentacja chunka

Prezentacją chunka nazwiemy:

- pierwsze się jego pojawienie,
- dodanie do pamięci deklaratywnej chunka już tam istniejącego – wtedy stary i nowy chunk zostaną połączone i zostanie odnotowana nowa prezentacja.

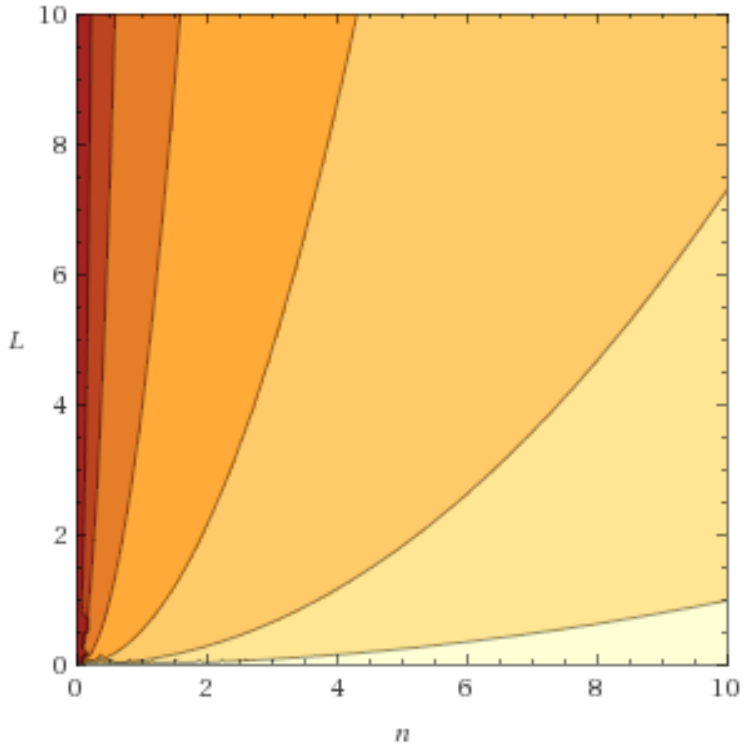
Uproszczony bazowy poziom aktywacji

Uproszczony bazowy poziom aktywacji zakłada, iż kolejne prezentacje chunka w przeszłości odbywały się w regularnych odstępach czasu. Uproszczony bazowy poziom aktywacji chunka, liczymy ze wzoru:

$$B_i = ln(\frac{n}{1-d}) - d \cdot ln(L),$$

gdzie:

- n to liczba „prezentacji" chunka i,
- L to czas od pierwszej prezentacji,
- d to parametr zaniku. Domyślnie wynosi 0.5, można go zmienić (*:bll*).



Szum (ϵ)

Szum (ϵ) składa się z dwóch składowych: szumu stałego i szumu chwilowego. Liczone są one podobnie, wartość szumu stałego jest jednak liczona tylko raz (przy utworzeniu chunka) a wartość szumu chwilowego za każdym obliczaniem na nowo.

Będziemy zazwyczaj pomijać szum stały i używać jedynie szumu chwilowego.

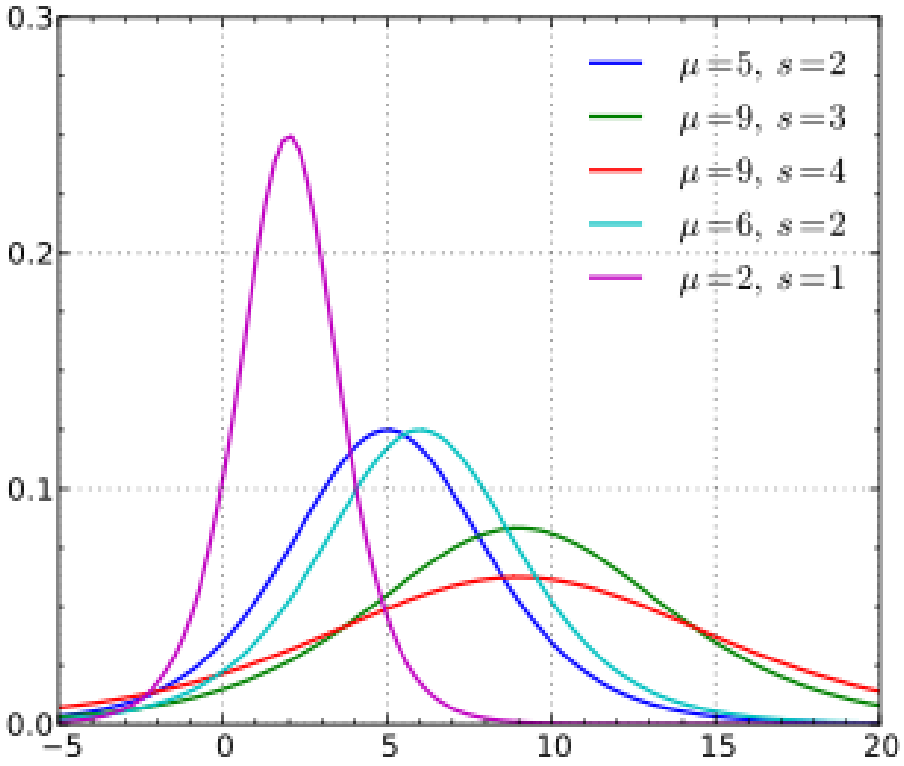
Szum (ϵ)

Szumy są losowane rozkładu logistycznego, ze średnią równą 0 i wariancją σ^2 zależną od parametru s:

$$\sigma^2 = \frac{\pi^2}{3} s^2$$

Wartość s dla szumu chwilowego będziemy ustawiać poprzez parametr *:ans* (dla szumu stałego *:pas*)

```
(sgp :ans 0.4)
```

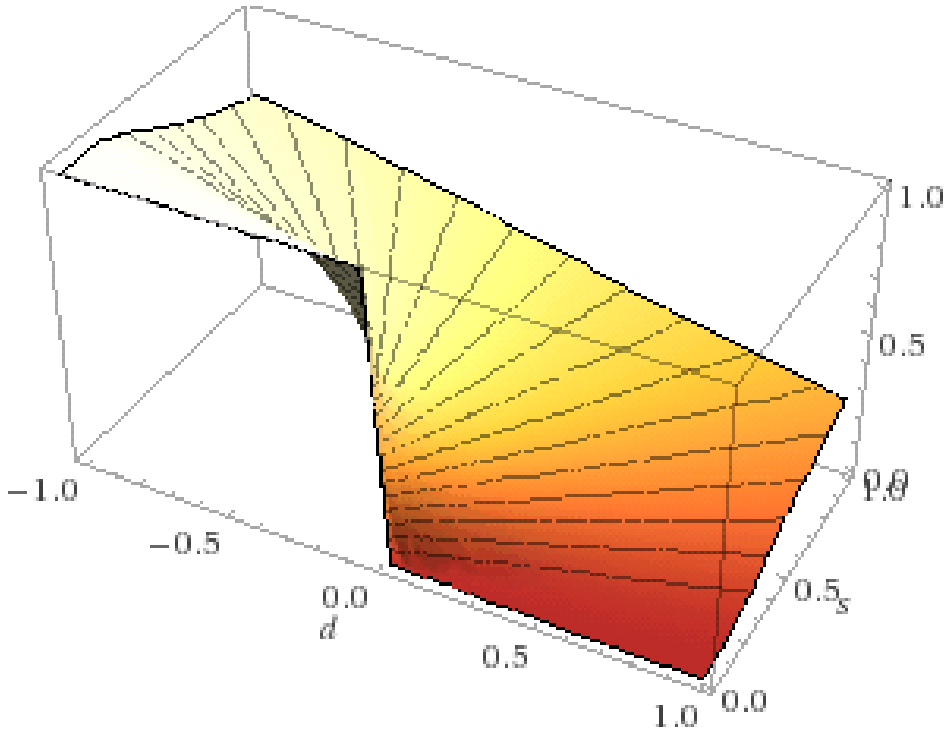


Rysunek: Gęstość prawdopodobieństwa dla rozkładu logistycznego.

Prawdopodobieństwo przekroczenia progu aktywacji

Prawdopodobieństwo przekroczenia progu aktywacji przez chunk i zależy od przedstawionych wcześniej wzorów i wynosi:

$$recallprobability_i = \frac{1}{1 + e^{\frac{\tau - A_i}{s}}}$$



Rysunek: $recallprobability_i, d = \tau - A_i$

Opóźnienia związane z odzyskaniem chunka z pamięci

Opóźnienia związane ze wstawieniem chunka do bufora *retrieval* również zależą od poziomu aktywacji A i wynoszą:

$$Time = Fe^{-A},$$

gdzie:

- A to poziom aktywacji odzyskiwanego chunka,
- F to współczynnik opóźnienia (parametr *:lf*).

Jeżeli żaden chunk nie zostanie odzyskany (i bufor *retrieval* zgłosi błąd), to opóźnienie wyniesie:

$$Time = Fe^{-\tau}.$$

Wiele pasujących chunków

Jeśli więcej niż jeden chunk pasuje do zapytania skierowanego do bufora *retrieval*, wybrany zostanie ten który ma najwyższą wartość funkcji aktywacji A (jeśli $A > \tau$).

Jako że A jest sumą deterministycznej wartości B i szumu ϵ, nie zawsze musi zostać wybrany ten chunk, którego bazowa wartość aktywacji (B) jest najwyższa!

Porównanie pełnego i uproszczonego wzoru na aktywację bazową

Zadanie 1

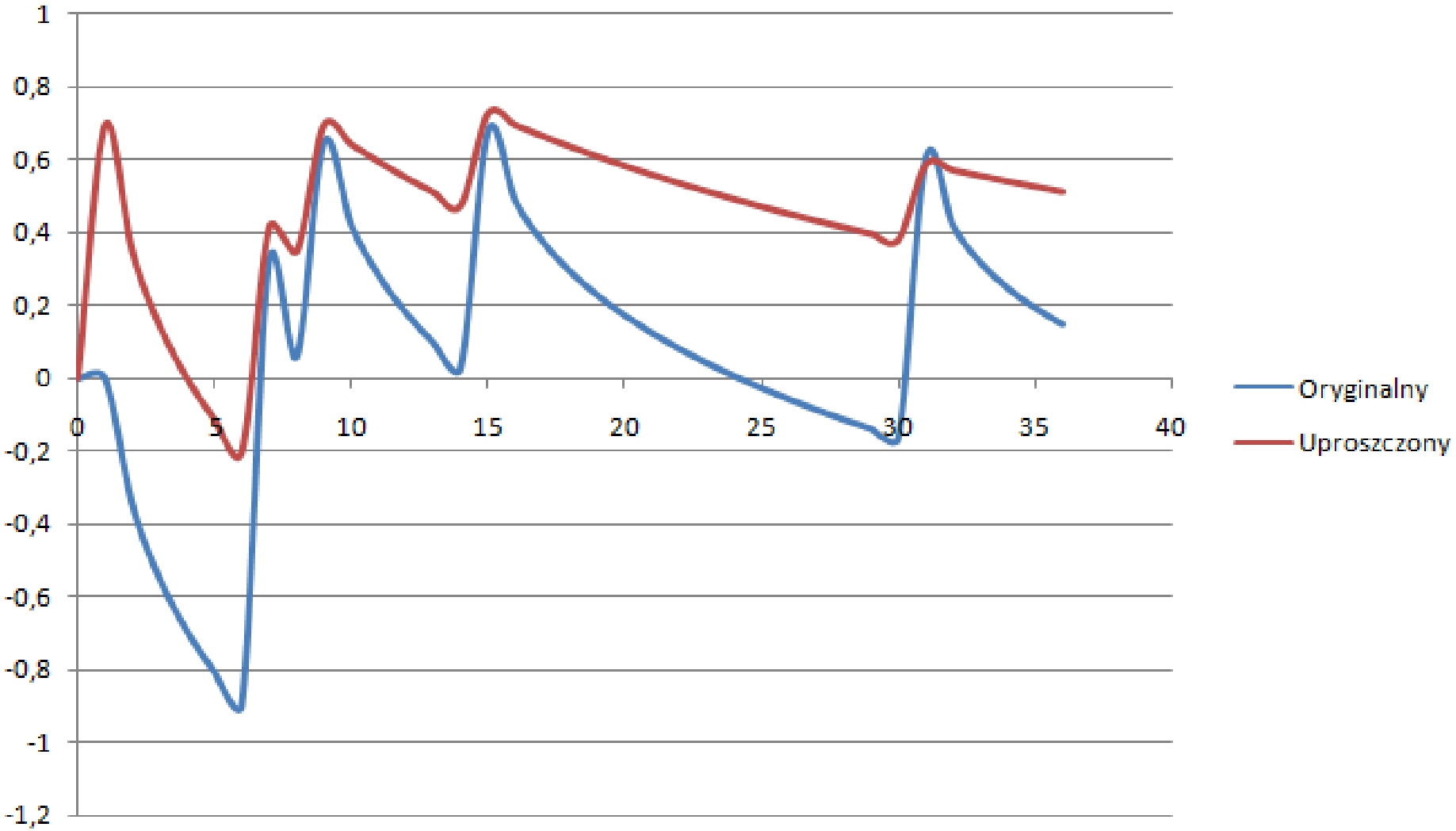
Utwórz w Excelu wykresy wartości bazowego poziomu aktywacji chunka X w czasie, jeśli jego prezentacje miały w miejsce w chwilach czasowych $t \in \{0, 6, 8, 14, 30\}$. Przyjmij jednostkowy krok czasowy.

a) Utwórz wykres prezentujący poprawną wartość bazowego poziomu aktywacji w czasie.

b) Utwórz wykres prezentujący uproszczoną wartość bazowego poziomu aktywacji w czasie.

c) Poeksperymentuj z chwilami czasowymi prezentacji chunka. Jaki mają one wpływ na podobienstwo przebiegów oryginalnego i uproszczonego? Kiedy oba przebiegi przyjmują podobne, a kiedy różne wartości?

Porównanie pełnego i uproszczonego wzoru na aktywację bazową



Zapamiętywanie par

Zadanie 2

Załaduj do ACT-R eksperyment *paired.lisp* i uruchom go za pomocą polecenia:
(paired-task m n)
gdzie *m* to liczba par (max 20) a *n* to liczba powtórzeń wewnatrz eksperymentu. Możesz równiez użyć polecenia *(paired-task m n 'human)* aby uruchomić eksperyment w trybie interakcji z użytkownikiem. Twoim celem jest wciśnięcie cyfry związanej w wyświetlanym słowem.

Plik z modelem:
...\ACT-R Standalone Environment\tutorial\unit4\paired.lisp

Zapamiętywanie par

Zadanie 3

Wybierz opcję „Production graph” i przeanalizuj strukturę modelu. W jakiej kolejności uruchamiane będą reguły? Która ścieżka była najczęściej używana? Dlaczego?

Spróbuj przeanalizować strukturę również innych modeli (np. model eksperymentu Sperlinga z poprzednich zajęć).

Krok 1: Czekaj na pojawienie się nowego słowa

```
(P attend-probe
=goal>
   ISA goal
   state start
=visual-location>
?visual>
   state free
==>
+visual>
   cmd move-attention
   screen-pos =visual-location
=goal>
   state attending-probe
)
```

Krok 2: Odczytaj słowo

```
(P read-probe
=goal>
   ISA goal
   state attending-probe

=visual>
   ISA visual-object
   value =val
?imaginal>
   state free
==>
+imaginal>
   ISA pair
   probe =val

+retrieval>
   ISA pair
   probe =val

=goal>
   state testing
)
```

Chunki par

Informacje o parach słowo - cyfra przechowywać będziemy w chunkach analogicznych do poniższego:

```
PAIR1
      probe   cat
      answer  7
```

```
+imaginal>
   ISA pair
   probe =val
```

W powyższych linijkach tworzymy nowy chunk w buforze *imaginal*. Póki co zawiera on jedynie slot *probe* – slot *answer* zostanie dopiero dodany w momencie odczytania z ekranu powiązanej z obecnym słowem cyfry.

Próba przypomnienia sobie powiązanej cyfry

```
+retrieval>
   ISA pair
   probe =val
```

W powyższych linijkach prosimy moduł deklaratywny o przypomnienie sobie chunka zawierającego potrzebną nam parę słowo - cyfra. Możliwe jest, iż w momencie wykonania powyższych lini chunk taki jeszcze nie istnieje. W takim wypadku, bufor *retrieval* zwróci błąd.

Zauważ! Potrzebne są nam oba zapytania do buforów *imaginal* i *retrieval*!

Jedno z nich tworzy nowy chunk który potrzebny jest do zapamiętania pary (bądź odświeżenia sobie informacji o niej), drugi próbuje przypomnieć sobie odpowiedź na podstawie posiadanej już wiedzy.

Krok 3a: Przypomnij sobie powiązaną wartość

```
(P recall
=goal>
   ISA goal
   state testing

=retrieval>
   ISA pair
   answer =ans

?manual>
   state free

?visual>
   state free

==>
+manual>
   cmd press-key
   key =ans

=goal>
   state read-study-item

+visual>
   cmd clear
)
```

Krok 3b: NIE przypomnij sobie powiązanej wartości

```
(P cannot-recall
=goal>
   ISA goal
   state testing

?retrieval>
   buffer failure

?visual>
   state free

==>
=goal>
   state read-study-item

+visual>
   cmd clear
)
```

Istnieją dwie możliwe przyczyny z których może zaistnieć `buffer failure`:

a) szukany przez nas chunk nie istnieje,

b) szukany przez nas chunk nie osiągnął wystarczającej wartości aktywacji ($A < \tau$).

Krok 4: Czekaj na pojawienie się cyfry

```
(P detect-study-item
=goal>
   ISA goal
   state read-study-item

=visual-location>

?visual>
   state free
==>
+visual>
   cmd move-attention
   screen-pos =visual-location

=goal>
   state attending-target
)
```

Krok 5: Zapamiętaj powiązanie słowa z cyfrą

```
(P associate
=goal>
   ISA goal
   state attending-target

=visual>
   ISA visual-object
   value =val

=imaginal>
   ISA pair
   probe =probe

?visual>
   state free

==>
=imaginal>
   answer =val

-imaginal>

=goal>
   state start

+visual>
   cmd clear
)
```

Zapamiętywanie poprzez usuwanie

```
=imaginal>
   answer =val
-imaginal>
```

Do chunka znajdującego się w buforze *imaginal* dodajemy slot *answer* z wartością przechowywaną w zmiennej *=val*. W tym momencie jest on nierozróżnialny od chunka przechowującego tą samą wiedzę w module *declarative* (o ile taki chunk istnieje).
Czyszcząc bufor *imaginal*, przenosimy nasz chunk do pamięci deklaratywnej gdzie:

a) zostaje on dodany do pamięci deklaratywnej (jeśli to jego pierwsze wystąpienie),

b) dodaje on jedną „prezentację" do już istniejącego, analogicznego chunka.

Wartości parametrów

```
(sgp :v nil :esc t :rt -2 :lf 0.4 :ans 0.5 :bll 0.5 :act nil :ncnar nil)
```

- Próg aktywacji τ?
- Parametr zaniku d?
- Parametr chwilowego zaszumienia s?
- Współczynnik opóźnienia F?

Wartości parametrów

(sgp :v nil :esc t :rt -2 :lf 0.4 :ans 0.5 :bll 0.5 :act nil :ncnar nil)

- Próg aktywacji τ? **-2**
- Parametr zaniku d?
- Parametr chwilowego zaszumienia s?
- Współczynnik opóźnienia F?

Wartości parametrów

(sgp :v nil :esc t :rt -2 :lf 0.4 :ans 0.5 :bll 0.5 :act nil :ncnar nil)

- Próg aktywacji τ? **-2**
- Parametr zaniku d? **0.5**
- Parametr chwilowego zaszumienia s?
- Współczynnik opóźnienia F?

Wartości parametrów

```
(sgp :v nil :esc t :rt -2 :lf 0.4 :ans 0.5 :bll 0.5 :act nil :ncnar nil)
```

- Próg aktywacji τ? **-2**
- Parametr zaniku d? **0.5**
- Parametr chwilowego zaszumienia s? **0.5**
- Współczynnik opóźnienia F?

Wartości parametrów

```
(sgp :v nil :esc t :rt -2 :lf 0.4 :ans 0.5 :bll 0.5 :act nil :ncnar nil)
```

- Próg aktywacji τ? **-2**
- Parametr zaniku d? **0.5**
- Parametr chwilowego zaszumienia s? **0.5**
- Współczynnik opóźnienia F? **0.4**

Dla chętnych – co zostało jeszcze do poznania

Unit 5 Kontekst, rozprzestrzenianie aktywacji

Unit 6 Uczenie się przydatności (*utility*) reguł

Unit 7 Uczenie się nowych reguł produkcji

Unit 8 Zaawansowane techniki w regułach produkcji