## Разрешаване на многозначността на частите на речта в текста със статистически методи

Би следвало вече да разбирате, че "*морфологичният анализ спрямо предварително зададен речник*" е вид обработка, която дава на изхода многозначни резултати. Многозначността (ambiguity) идва от факта, че словоформите функционират по многозначен начин като морфологични варианти на дадена лема (или значение). С други думи, *има повече морфологични варианти или значения, отколкото приемливи низове, и затова един низ може да е "натоварен" с повече от една морфологична или семантична роля*. Това не затруднява човека, но е сериозен проблем за компютъра. **Уводен пример:** низът КОСА се разпознава по поне 4 начина:
- съществително, ед. ч-ло, женски род, нечл.; уред на косене;
- съществително, ед. ч-ло, женски род, нечл.; част от тялото;
- съществително, ед. ч-ло, мъжки род, кратък член; форма на КОС (птица);
- съществително, ед. ч-ло, мъжки род, бройна форма; форма на КОС.

В този текст е показано как статистическите методи могат да се прилагат за разрешаване на многозначността при автоматично разпознаване на частите на речта в даден непознат текст. До края на курса ще видим различни статистическите методи за обработка на едноезичен или многоезичен текст, като при някои от тях преброяването в корпусите ще се извършва на ниво **символ**, т.е. *няма да се ползват каквито и да било лингвистично-значими единици*. Но сега ще започнем с един статистически метод, който ползва думите като текстови единици. Това е може би най-успешният и широко разпространен метод, който се прилага на етапа на морфологичния анализ с цел разрешаване на многозначността (disambiguation).

### 1. Предварителни понятия: Вероятности

**Вероятност на събитие** (probability): възможността то да се случи. Записваме като цели числа 1 или 0; като дроби: 0.1, 1/10; или като проценти 50%. **Случайна променлива** (random variable) варира над предварително зададено множество от стойности (в нашия случай крайно). Има функция PROB, която присвоява вероятност на всяка възможна стойност на случайната променлива, напр. ако $e_1, e_2, ...., e_n$ са възможните различни стойности на случайна променлива E, то за всяка функция PROB трябва да са изпълнени условията
1. $PROB(e_i) \geq 0$, за всяко *i*;
2. $PROB(e_i) \leq 1$, за всяко *i*;
3. $\sum_{i=1,...,n} PROB(e_i) = 1$.

Пример: Ако конят Вихър от 100 надбягвания печели 20 и губи 80, това може да се запише като PROB(R=Печалба)=0.2 и PROB(R=Загуба)=0.8, където R е случайна променлива с две стойности {Печалба, Загуба}.

Понеже може да има много случайни променливи за описание на свързани явления, ние се интересуваме как техните стойности се отнасят една към друга. Например, дали конят Вихър печели надбягвания винаги когато вали? Тази интуитивно съществуваща връзка се изразява чрез така наречената **условна вероятност** (conditional probability) PROB(e | e'), напр. PROB(Печалба| Вали). Условната вероятност се изчислява по формулата

PROB(e|e')=PROB(e & e') / PROB(e'),

където PROB(e & e') е вероятността двете събития е и е' да се случат заедно. Например ако знаем, че PROB(Вали)=0.3 и PROB(Печалба& Вали)=0.15, то изчисляваме, че PROB(Печалба | Вали)= 0.15/0.3= 0.5.

Теоремата на Бейс за условната вероятност ни помага в случаите, когато нямаме пълна информация за всички възможни случаи и зависимости:

PROB(A|B) = (PROB(B|A) ∗ PROB(A) ) / PROB(B).

Сега можем да изчислим вероятността да е валяло в деня, когато конят Вихър е спечелил надбягванията:

PROB(Вали|Печалба)=

(PROB(Печалба|Вали)∗PROB(Вали))/ PROB(Печалба) =

(0.5 ∗ 0.3) / 0.2 = 0.75.

Две събития A и B могат да бъдат **независими** (independent) едно от друго. Тогава PROB(A|B)=PROB(A) и съответно

PROB(A&B)=PROB(A) ∗ PROB(B).

Да минем към примери от естествения език, например разпознаване на думите от дадено изречение като части на речта. Този процес се нарича маркиране (**part-of-speech tagging**). Самите думи са многозначни **като части на речта,** например "бели" е форма на глагол, на съществително и прилагателно и може да бъде маркирано като категория N, V или Adj. Следователно задачата може да се постави по следния начин: *за дадено изречение, да се определи за всяка негова дума най-вероятната част на речта, към чиято категория тези дума принадлежи*. Разглеждаме елементарни случаи, за да илюстрираме основните идеи: нека си представим, че задачата ни е да идентифицираме коректната синтактична категория за думи, които могат да бъдат или съществителни N, или глаголи V. Тази задача се формализира чрез две случайни променливи: C (от категория), със стойности N или V, и W (от дума), която има за стойности всички възможни думи. Тогава нашата задача се свежда до определяне например дали PROB(C=N | W=flies) е по-голяма от PROB(C=V |

W=flies), за да можем да заключим, че думата flies е съществително. Можем да изпускаме имената на случайните променливи от формулите, и тогава нашите формули се редуцират до

PROB( N | flies) = PROB( flies & N) / PROB(flies).
PROB( V | flies) = PROB( flies & V) / PROB(flies).

Сега очевидно стигаме до проблема как да определим по-голямата от двете вероятности PROB( flies & N) и PROB( flies & V), понеже делителят е еднакъв и в двете формули.

Как да определим обаче тези две вероятности? Нямаме запис на всички възможни текстове, за да ги измерим точно, но при голямо количество данни можем поне да ги приближим в желаната степен. Да си представим корпус от 1 273 000 думи, само с прости изречения, в които flies се среща 1000 пъти, от които 400 пъти като съществително и 600 пъти като глагол. Тогава:

PROB(flies) ~= 1000/1273000 = 0.0008, приемаме го за апроксимация на честотата на думата flies изобщо, макар и над корпус от 1 273 000 думи; и

PROB(flies & N) ~= 400 / 1 273 000 = 0.0003
PROB(flies & V) ~= 600 / 1 273 000 = 0.0005

Тогава по-вероятно е flies да е глагол, а не съществително. Тази условна вероятност е: PROB(V|flies)=

PROB(flies & V) / PROB(flies) =
0.0005 / 0.0008 = 0.625.

Очевидно нашият метод ще работи коректно в около 60% от случаите, *но все пак това е по-добре, отколкото всеки път да предполагаме, че flies е съществително*. Тази идея ще бъде доразвита по-долу. Тук тя е изложена много елементарно, понеже думите се разглеждат като изолирани събития в текста. Забележете, че макар точността на метода да ни е известна: ~60%, **ние не можем да разграничим верните от грешните разпознавания**.

## 2. Оценка на вероятностите

Ако имаме всички данни свързани с дадена задача, можем да изчислим точно вероятностите на събитията в задачата. Например, ако конят Вихър е участвал само в 100 състезания, виждаме веднага с каква вероятност той печели в тях. Но това не е начинът, по който искаме да използваме вероятностите - целта ни е да предскажем шансовете му за победа на 101-вото състезание. Подобно е положението с ЕЕ; ние искаме да анализираме текстове, които не са ни предварително известни. Тогава използваме данните от анализираните вече изречения, за да предскажем интерпретацията на следващото изречение. Работим главно с приближени вероятности (няма как да знаем истинските). По-горе видяхме един пример на оценка: ако от 1000 срещания на flies 600 са като глагол, то

PROB(V|flies)=0.6 и предположението се прилага към 1001-вото срещане. Това просто съотношение се нарича оценка на максималното подобие (**maximum likelihood estimator, MLE**). При достатъчно много изходни данни MLE е достатъчно добро приближение на истинската вероятност.

По закона за големите числа, при неограничени данни истинската вероятност може да бъде приближена с каквато точност искаме. Ако се вземе малък брой примери обаче, се получава незадоволително приближение. Пример: опитваме се да приближим истинската вероятност на събитието една подхвърлена (идеална) монета да падне Ези или Тура. Ние знаем, че истинската вероятност е 0.5, и нека за целите на задачата приближението е достатъчно добро, когато е в интервала [0.25, 0.75]. Това е **интервалът на допустима грешка** (margin of error). При две хвърляния, има 4 възможни изхода, показани по-долу на таблица 1:

Таблица 1:

| Резултат от двете хвърляния: | Вероятност да се падне ези PROB(E): | Допустимо приближение ли е? |
|---|---|---|
| EE | 1.0 | Не |
| ET | 0.5 | Да |
| TE | 0.5 | Да |
| TT | 0 | Не |

При 3 хвърляния има 8 възможни изхода и картината се променя, таблица 2:

| Резултат от три хвърляния: | Вероятност да се падне ези PROB(E): | Допустимо приближение ли е? |
|---|---|---|
| EEE | 1.0 | Не |
| EET | 0.66 | Да |
| ETE | 0.66 | Да |
| ETT | 0.33 | Да |
| TEE | 0.66 | Да |
| TET | 0.33 | Да |
| TTE | 0.33 | Да |
| TTT | 0 | Не |

Както се вижда на Таблица 1, при 2 хвърляния сме постигнали достатъчно добро приближение на истинската вероятност при 50% от случаите; при 3 хвърляния - в 75% от случаите (таблица 2); при 4 ще бъде в 87.5% от случаите, при 8 - в 93% от случаите, при 12 - в 95% и т.н. При достатъчно голям брой опити може да се постигне колкото искаме добро "допустимо приближение на истинската вероятност". При обработката на ЕЕ това означава, че вземаме все по-големи учебни корпуси - *докато резултатите по обработка на новия текст спрат да се влияят от големината на корпуса.* В този момент спираме, понеже повече не можем да подобрим системата спрямо въпросния метод (а само с принципни поправки).

Колкото и да е добър методът за приближение на истинската вероятност обаче, при ЕЕ имаме следните особености:
а) трябва да се направят най-различни приближения, спрямо вероятностите на различни събития моделиращи явления на различни ЕЕ-нива;
б) голяма част от събитията са съвсем редки. Това е проблемът на "разпръснатите" (?) данни (**sparse data**). Например, един известен текстов корпус (the Brown corpus) съдържа около 1 млн. думи (словоупотреби), но това са всъщност само 40 000 различни думи, които се повтарят многократно. Можем да си помислим, че те се повтарят средно по около 25 пъти, но това не е вярно. Повечето от тези 40 000 думи се срещат под 5 пъти. При нашата така поставена задача - да разпознаем автоматично частите на речта - по-редките думи (въпросните 38 000) са в много неизгодно положение. Още повече, редица от тях не се срещат във всички свои словоформи и тогава вероятността на тези словоформи е 0.

Да навлезем малко в техниките за приближена оценка на вероятността на събитията с ниска честота. Първо ще въведем рамка, в която такива събития могат да бъдат сравнявани. Ако имаме случайна променлива X, всички техники използват множество от стойности $V_i=\{x_1,x_2,...x_n\}$, конструирано чрез наблюдения колко пъти X взема конкретна стойност (това са стойностите $x_i$). Например може да се използва $V_i = |x_i|$, т.е. $V_i$ е бройката колко пъти $X = x_i$. След като $V_i$ се определи за всяко $x_i$, приближените вероятности се получават по формулата

$$\mathrm{PROB}(X = x_i) \sim= V_i / \sum_i V_i, \qquad (\textit{fele})$$

където знаменателят гарантира трите свойства на PROB от стр.1 по-горе.

Един начин за избягване на вероятност 0 е да се добавя малко число към всеки брой срещания, напр. 0.5, т.е. $V_i = |x_i|+0.5$. Тази гаранция за не-нулева вероятност запазва приликата между често срещащите се стойности и редките такива. Тази техника се нарича оценка на очакваното подобие (?) (**expected likelihood estimator, ELE**). Да видим разликата между MLE и ELE: нека имаме дума **w,** която не се среща в копруса, и нека разгледаме приближена оценка на вероятността **w** да е в един от 36-те граматически класа $L_1$, $L_2$, ..., $L_{36}$ описани по-долу в таблица 3. Тогава имаме случайна променлива X, за която $X=x_i$ само когато **w** е в граматическата категория $L_i$. Тогава MLE за $\mathrm{PROB}(X=x_i)$ няма да бъде дефинирано заради знаменател 0. Обаче ELE дава равна вероятност на 36-те класа; от $V_i = |x_i|+0.5$ всяко $V_i$ ще бъде 0.5 и така по формула (*fele*): $\mathrm{PROB}(L_i \mid w) \sim= 0.5 / 18 = 0.0277$, т.е. 1/36. Така ELE отразява по-добре факта, че не знаем нищо за думата.

Но за сметка на това при думи с ненулеви срещания ELE е много "консервативно". Ако w се среща в корпуса 5 пъти, веднъж като глагол и 4 пъти като съществително, тогава по MLE имаме $\mathrm{PROB}(N \mid w) = 0.8$, а по

ELE по формула (*fele*): PROB(N | w) = 4.5 / (17 +4.5 + 1.5) = 4.5/23 = 0.1956, което не съответства на интуитивното ни усещане за честота.

Какво се прави на практика? Формулира се задача и се избират подходящи случайни величини за моделирането й. Взема се голям корпус, подходящо маркиран ча целта, и част от него се използва като **training data** (**training set**). Около 10-20% от данните се използват за **test set**. Обикновено двете части не се пресичат. Най-удачно е след разработване на алгоритмите да се направят няколко опита, като различни "парчета" от корпуса се слагат и махат като **training** и **test.** Така можем да сме сигурни, че при еднакво добри резултати за различни парчета, сме постигнали добър формален модел на задачата (от една страна) и напълно сме отразили особеностите на данните (от друга страна). Тук задължително трябва да отбележим голямата свобода, която е характерна за задачи от този тип: както ще видим по-долу, при разпознаване частите на речта имаме различна възможност за избор на маркери. Така че има смисъл от повече опити над различни корпуси.

### 3. Part-of–Speech Tagging (автоматично маркиране на частите на речта)

Нека разгледаме следното множество маркери на POS-категории за английския език, което се използва в проекта Penn Treebank (таблица 3):

| No | Маркер | Категория | No | Маркер | Категория |
|---|---|---|---|---|---|
| 1. | CC | Coordinating Conjunction | 19. | PP$ | Possessive Pronoun |
| 2. | CD | Cardinal Number | 20. | RB | Adverb |
| 3. | DT | Determiner | 21. | RBR | Comparative Adverb |
| 4. | EX | Existensial *there* | 22. | RBS | Superlative Adverb |
| 5. | FW | Foreign Word | 23. | RP | Particle |
| 6. | IN | Preposition / subord. conj | 24. | SYM | Symbol (math, scientific) |
| 7. | JJ | Adjective | 25. | TO | to |
| 8. | JJR | Comparative Adjective | 26. | UH | Interjection |
| 9. | JJS | Superlative adjective | 27. | VB | Verb base form |
| 10. | LS | List Item markers | 28. | VBD | Verb, past tense |
| 11. | MD | Modal | 29. | VBG | Verb, gerund/pres.partic |
| 12. | NN | Noun, singular or mass | 30. | VBN | Verb, past participle |
| 13. | NNS | Noun, plural | 31. | VBP | Verb, non-3s, present |
| 14. | NNP | Proper noun, singular | 32. | VBZ | Verb, 3s, present |
| 15. | NNPS | Proper noun, plural | 33. | WDT | Wh-determiner |
| 16. | PDT | Predeterminer | 34. | WP | Wh-pronoun |
| 17. | POS | Possessive ending | 35. | WPZ | Possessive wh-pronoun |
| 18. | PRP | Personal pronoun | 36. | WRB | Wh-adverb |

На всяка дума в дадено изречение трябва да се присвои един от горните (най-вероятния) маркер. Вече видяхме най-простия алгоритъм в част 1:

винаги вземай най-често срещащата се стойност в избраното training set. За наша изненада, този алгоритъм често работи за над 90% от думите, понеже повече от половината думи в един корпус не са многозначни. Така че 90% са основата, от която тръгваме към подобрения. Един статистически метод работи добре, когато той постига точност над 95-96%. (Счита се, че при ръчно маркиране човекът би направил също известен процент грешки).

Възможни са усъвършенствания чрез разглеждането на контекста на думата. Например, ако flies се предшества от the, значи е съществително. Нека $w_1$, $w_2$, ..., $w_T$ е поредица от думи. Искаме да намерим поредицата от лексикални категории $C_1$,$C_2$,...,$C_T$ която максимизира

1. PROB(C1,C2,...CT | w1, ..., wT).

Да се генерира приближена оценка на вероятността на много данни би отнело обаче твърде много време; (1) се преформулира по формулата на Бейс в еквивалентното:

2. (PROB(C1,C2,...CT) ∗ PROB(w1, ..., wT | C1,C2,...CT) ) / PROB(w1, ..., wT)

Така нещата се свеждат до намиране на редица C1,C2,...CT, която максимизира числителя на дробта (2), а именно

3. PROB(C1,C2,...CT) ∗ PROB(w1, ..., wT | C1,C2,...CT)

Тези дълги редици от категории са също много трудни за обработване, но вероятността им може да бъде апроксимирана чрез по-прости вероятности, ако се предположи известна независимост. И двата израза в (3) се апроксимират. Първият израз, вероятността на редица категории, може да бъде апроксимиран чрез серии от вероятности, изчислени на базата на ограничен брой предишни категории. Най-често се използват една или две предишни категории. Моделът на биграмите (bigram model) преглежда двойките категории (или думи) и използва условната вероятност PROB(Ci|Ci-1), че категорията Ci ще се случи след категорията Ci-1. Моделът на триграмите (trigram model) следи две минали категории и се основава на PROB(Ci|Ci-1 Ci-2). Триграмите дават по-добър резултат на практика, но тука за простота се спираме на биграмите. Използваме приближението:

$$\text{PROB}(C1,C2,...CT) \sim= \prod_{i=1,...,T} \text{PROB}\,(Ci|Ci\text{-}1).$$

Маркираме началото на изречението (позиция 0) с псевдокатегорията ∅ като стойност на C0. Така първата биграма на изречение, започващо с ART, ще бъде PROB(ART|∅). Сега например изречение с ART N V N ще има приближена вероятност, изчислена по формулата

PROB(ART N V N) ∼=

PROB (ART|∅) ∗ PROB (N|ART) ∗ PROB (V|N) ∗ PROB (N|V)

Втората вероятност - множител в (3), - PROB(w1, ..., wT | C1,C2,...CT) се опростява при предположението, че една дума принадлежи към дадена категория независимо от думите в предишните категории. Така имаме

PROB(w1, ..., wT | C1,C2,...CT) ~= $\Pi_{i=T}$ PROB(wi | Ci).

Сега вече (3) се свежда до намиране на редица от категории C1,C2,...CT, която максимизира

(4) $\Pi_{i=1,...,T}$ PROB (Ci|Ci-1) * $\Pi_{i=T}$ PROB(wi | Ci).

Тези вероятности в (4) могат наистина да бъдат изчислени приближено от "учебен" текстов корпус, в който всяка дума е маркирана с нейната правилна част на речта. В частност, по дадена база от текстове, вероятностите-биграми могат да се приближат просто като се преброи колко пъти се среща всяка двойка от категории като сравнение с броя пъти на срещане на индивидуалните категории. Например, ето приближение на вероятността един глагол V да следва N:

(5) PROB(Ci=V | Ci–1=N) ~= Броя на (N в позиция i-1 и V в позиция i) /
Броя на (N в позиция i-1 и не-V в позиция i)

По тази формула са изчислени стойностите в последния стълб на долната таблица 4. Тя се отнася за **изкуствено генериран учебен корпус** на английски език - *опростен и подходящ за примери в нашата лекция* - от

- 300 изречения, което значи 300 празни категории начало;
- 4 маркера на категории (N, V, ART, P - забележете данните от реалния корпус в табл. 3 са с **36 категории**, но ние вземаме прост пример),
- 1998 думи (833 N, 300 V, 558 ART, 307 Prepositions).

Биграмите са приближени по формула (5). За решаване на проблема с разпръснатите данни (sparse data), всяка непоказана в таблица 4 биграма има присвоена по премълчаване вероятност 0.0001.

Табл. 4: Пресмятане на вероятността за срещане на биграми в учебния корпус (последните два стълба са графично илюстрирани на фигура 2):

| Категория | Общ брой срещания | Двойка категории | #срещания като двойка | Биграма | Прибл. вероятн |
|---|---|---|---|---|---|
| ∅ | 300 | ∅, ART | 213 | PROB(ART\|∅) | 0.71 |
| ∅ | 300 | ∅, N | 87 | PROB(N\|∅) | 0.29 |
| ART | 558 | ART,N | 558 | PROB(N\|ART) | 1 |
| N | 833 | N, V | 358 | PROB(V\|N) | 0.43 |
| N | 833 | N, N | 108 | PROB(N\|N) | 0.13 |
| N | 833 | N, P | 366 | PROB(P\|N) | 0.44 |
| V | 300 | V, N | 75 | PROB(N\|V) | 0.35 |
| V | 300 | V, ART | 194 | PROB(ART\|V) | 0.65 |
| P | 307 | P, ART | 226 | PROB(ART\|P) | 0.74 |
| P | 307 | P, N | 81 | PROB(N\|P) | 0.26 |

Вероятностите за лексикалното срещане, т.е. PROB(wi|Ci), могат да се пресметнат чрез преброяване коя дума по колко пъти се среща в дадена категория. Таблица 5 по-долу дава примери:

| **Дума** | **N** | **V** | **ART** | **P** | **Общо** |
|---|---|---|---|---|---|
| flies | 21 | 23 | 0 | 0 | 44 |
| fruit | 49 | 5 | 0 | 0 | 55 |
| like | 10 | 30 | 0 | 21 | 61 |
| a | 0 | 0 | 202 | 0 | 202 |
| the | 0 | 0 | 303 | 0 | 303 |
| flower | 53 | 15 | 0 | 0 | 68 |
| flowers | 42 | 16 | 0 | 0 | 58 |
| birds | 64 | 1 | 0 | 0 | 65 |
| други думи ... | 592 | 210 | 53 | 286 | 1142 |
| Общо | 833 | 300 | 558 | 307 | 1998 |

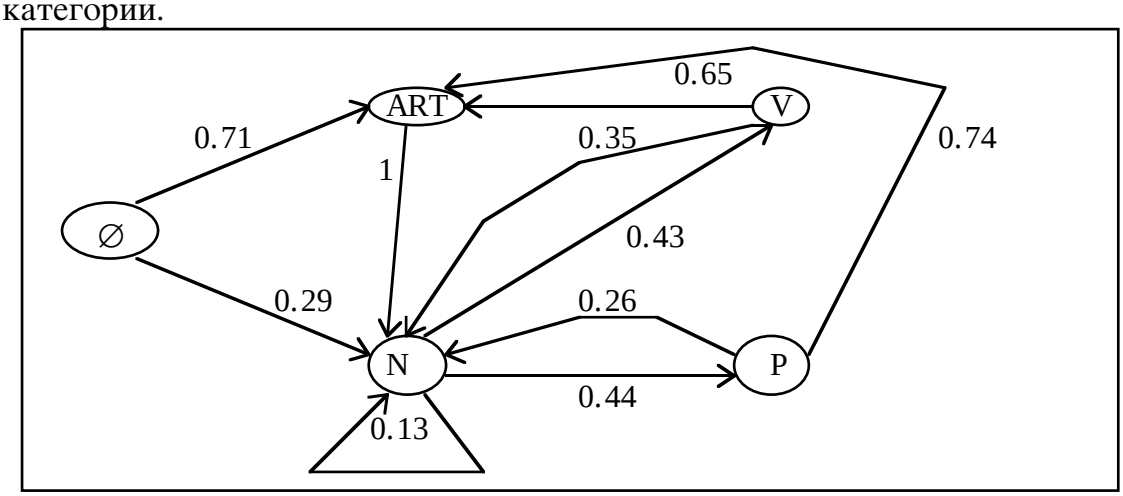
Да забележим, че всъщност се брои коя категория чрез каква дума се реализира. Например, вероятността PROB(the|ART) се изчислява като Брой(#пъти the е ART)/ Брой(#пъти има ART). Другата вероятност, PROB(ART|the) е съвсем различна стойност. Ето на фиг. 1 някои стойности на PROB(wi|Ci):

| | |
|---|---|
| PROB(the\|ART) = 0.54 | PROB(a\|ART) = 0.36 |
| PROB(flies\|N) = 0.025 | PROB(birds\|N) = 0.076 |
| PROB(flies\|V) = 0.076 | PROB(flower \|N) = 0.063 |
| PROB(like\|V) = 0.1 | PROB(flower \|V) = 0.05 |
| PROB(like\|P) = 0.068 | PROB(flowers \|N) = 0.05 |
| PROB(like\|N) = 0.012 | PROB(flowers \|V) = 0.053 |

**Фиг. 1.** Вероятности за лексикално срещане на думи в категории по учебния корпус

Сега, как можем да максимизираме (4)? При дадени M категории и T думи, има N**T възможности категориите да се реализират чрез тези думи. Но ние сме направили предположение за опростяване чрез биграмите: че категорията Ci зависи само от Ci-1. Сега процесът може да бъде моделиран чрез специални вероятностни крайни автомати, както е показано по-долу на фигура 2. Всеки връх е лексикална категория, а етикетът на стрелките показва вероятността двете категории да се срещнат една след друга. С такива машини може да се изчисли вероятността за последователно срещане на категории, като обхождаме стрелките и умножаваме вероятностите. Например поредица ART N V N има вероятност 0.71 ∗ 1 ∗ 0.43 ∗ 0,35 = 0.107. Подобна мрежа е смислена само ако вероятностите на стрелките са изчислени като вероятности едната категория да се срещне след другата. Това предположение се нарича **предположение на Марков** (Markov assumption), а мрежата на фиг. 2 - верига на Марков (**Markov chain**).

В този момент виждаме колко е важно да работим със сравнително малък набор от категории - около 40 - понеже сметките съществено зависят от броя на избраните лексикални категории C1, C2, ..., CT. Повечето съществуващи маркировъчни програми (taggers) работят с около 45 категории.




Фигура 2. Верига на Марков построена по биграмите на таблица 4.

Веригата на Марков може да бъде разширена с цел отразяване и на вероятностите за лексикални срещания от фигура 1. Към всеки връх може да бъде асоциирана изходна вероятност (output probability), която е вероятност на някакъв възможен изход, случващ се в този връх. Например, към N на фигура 2 можем да присвоим вероятността да бъде реализирана някоя дума като N; т.е. към N се присъединява запълнена таблица с данни като на фиг.1, която показва каква е вероятността N да бъде дума_1, дума_2, и т.н. за всички думи. Мрежа като показаната по-долу на фигура 3 се нарича **Hidden Markov Model** (HMM), като "скрит" означава, че за дадена поредица от думи не се знае в кое състояние се намира HMM за нея. Например, "flies" може да се генерира от състояние N с вероятност 0.025 или от състояние V с вероятност 0.076. Поради тази многозначност вече не е тривиално да се изчисли по мрежата вероятността на срещане на редица от думи. Ако обаче имаме конкретна редица от категории, можем лесно да изчислим вероятността тази редица да генерира конкретен изход; умножаваме "вероятността на пътя" по "вероятността да се породят тези думи". Например, веригата от фигура 2 генерира N V ART N с вероятност

(6) $0.29 * 0.43 * 0.65 * 1 = 0.081.$

За дадени конкретни думи "Flies like a flower" имаме

PROB(flies|N) * PROB(like|V) * PROB(a|ART) * PROB(flower|N) =

(7) 0.025 *1 * 0.36 * 0.063 = 5.4 * 10**(-5)

Умножаваме (6) по (7) и получаваме вероятността 4.37 * 10**(-6). Това е вероятността от машината на Фиг. 2 с вероятностите от Фиг. 1 да се генерира изречението “Flies like a flower”.

Сега да се върнем към нашата задача. Както вече решихме, приемлива формула за изчисление на вероятността дадена редица от думи w1, w2, ..., wT да принадлежат към редица от категории C1, C2, ..., CT е както следва:

(4) $\Pi_{i=1,...,T}$ PROB (Ci|Ci-1) * $\Pi_{i=T}$ PROB(wi | Ci).

Всъщност ние търсим най-вероятните категории, към които думите принадлежат в конкретната си употреба една след друга.

Сега да направим едно важно наблюдение: няма нужда да изчисляваме вероятностите на всички възможни редици от категории (те са експоненциален брой), а само на някои редици. Това се обяснява с предположението на Марков. Забелязваме, че под-редици от категории, които започват и завършват с една и съща категория могат да бъдат пропуснати от разглеждането: те всъщност оказват такова влияние на разглеждането на следващата вероятност, както една категория. Значи, в известен смисъл трябва да пазим *само редицата от най-вероятни категории спрямо последната категория и да игнорираме другите редици*.

Например: нека ни е дадено несрещаното до сега изречение

“Flies like a flower”

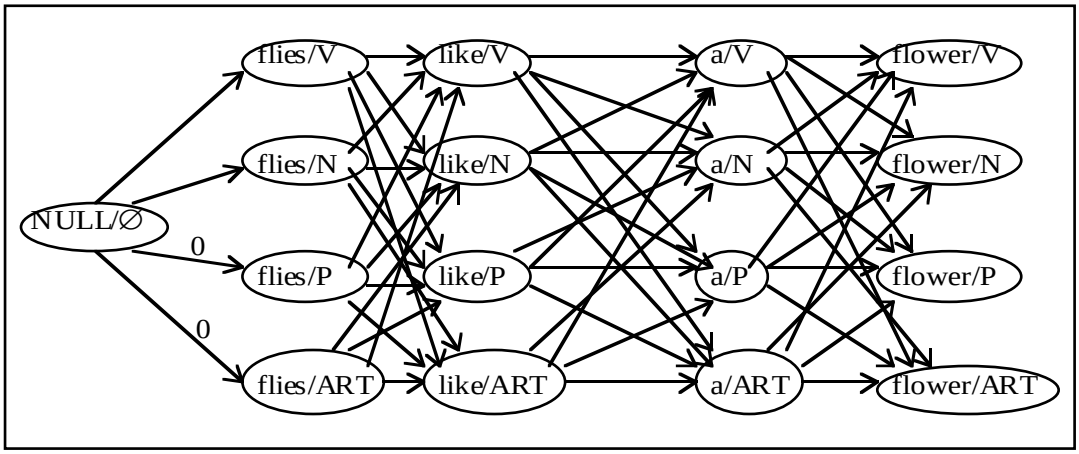
и да разгледаме задачата за намиране на най-вероятните категории за неговите думи.

За всяка дума знаем вероятните й лексикални срещания, дадени на фиг. 1. Научили сме ги от учебен корпус, в който частите на речта са били (ръчно) отбелязани и ние сме ги преброили. Знаем и биграмите за съвместни срещания на категориите, изчислени също над учебния корпус. Решили сме въпроса с непознатите думи и техните категории чрез присвояване на вероятности “по премълчаване”. Можем да започнем работа за това изречение с дължина 4 думи.

При 4 изобщо възможни лексикални категории (N,V,ART,P), има 4**4 възможни редици от категории, или 256. Търсим най-вероятната от тях. По метода на грубата сила би трябвало да генерираме всичките 256 редици и на изчислим техните вероятности. Не ни харесва. По предположението на Марков свиваме това множество до показаното на фигура 3. За всяка дума ще разглеждаме само възможностите да я следват 4 върха, по един за всяка категория. Сега мрежата на преходите от фиг. 3 е кодиране на всичките 256 възможни редици категории за даденото изречение. От тях са премахнати неадекватните. За да намерим най-вероятната редица с дължина 4, първо търсим най-вероятната редица с дължина 2, за да “покрием” думите “flies

like”. В света на фиг. 2 началото на изречението е ART или N. Значи в този момент изчисляваме 4 редици и вземаме най-добрата:

- за flies като N и like като N,
- за flies като N и like като V,
- за flies като N и like като ART,
- за flies като N и like като P.



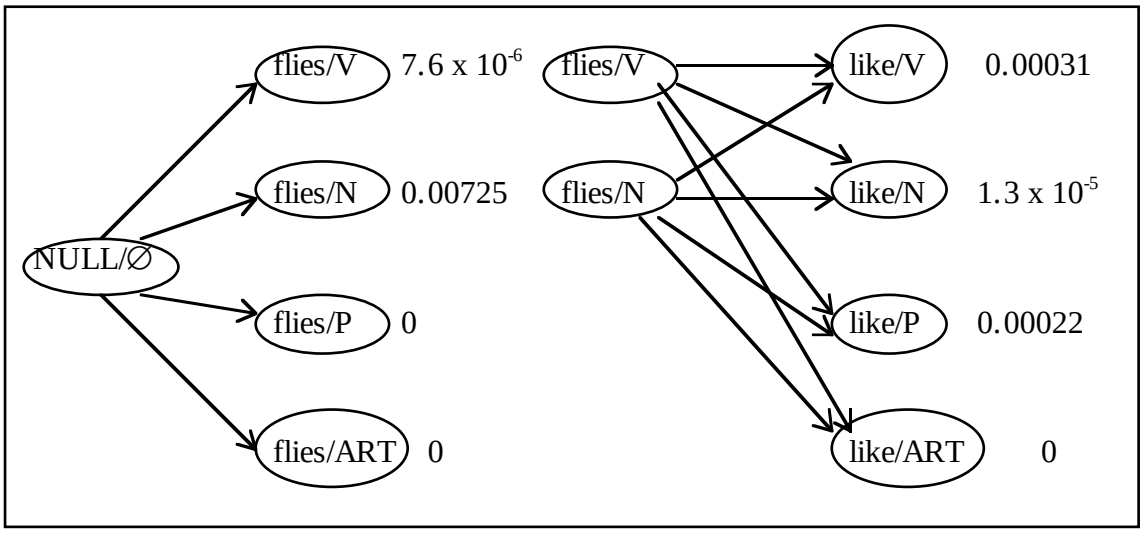

Фигура 3. Кодиране на 265 възможни последователности с използване на предположението на Марков.

След това използваме тази информация, за да изчислиме най-добрите 4 редици за “flies like a” с възможен край “a”:

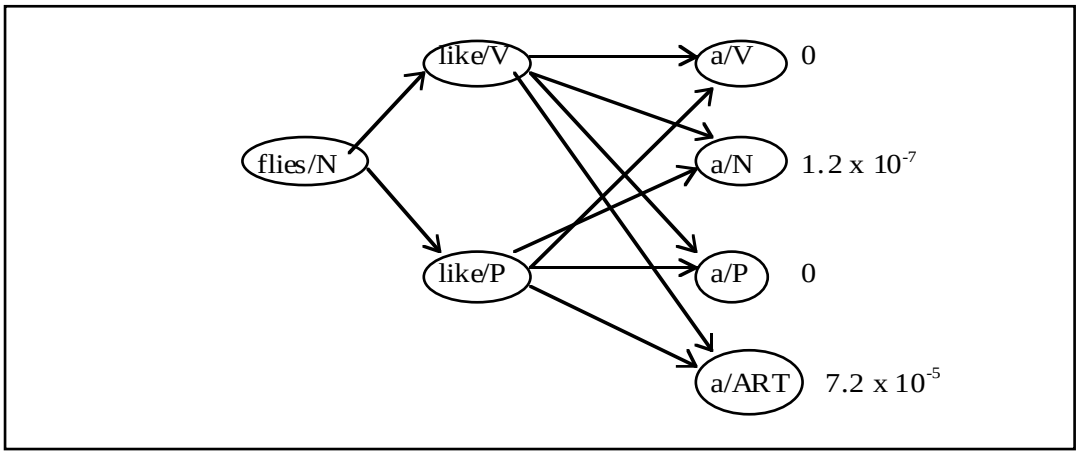
- “a” като N,
- “a” като ART,
- “a” като V,
- “a” като P.

Пак вземаме най-добрата и така до последната стъпка. От получените 4 редици на последната стъпка, вземам най-добрата и я присвояваме на цялото изречение. Този алгоритъм се нарича “алгоритъм на Витерби”. Резултатите му са показани на фигури 4, 5 и 6 (като са пресметнати повече от една редици за илюстрация). За T думи и N лексикални категории, той намира най-вероятната редица за k∗T∗N∗∗2 стъпки, за константа k значително по-добра от числото N∗∗T, изисквано при пълно претърсване. Нека запомним, че такива алгоритми са ефективни когато вероятностите са изчислени според наблюдения върху голям, ръчно маркиран корпус. “Наивното” предсказване води до точност под 90%, а POS–tagger-ите имат точност над 95-96%.

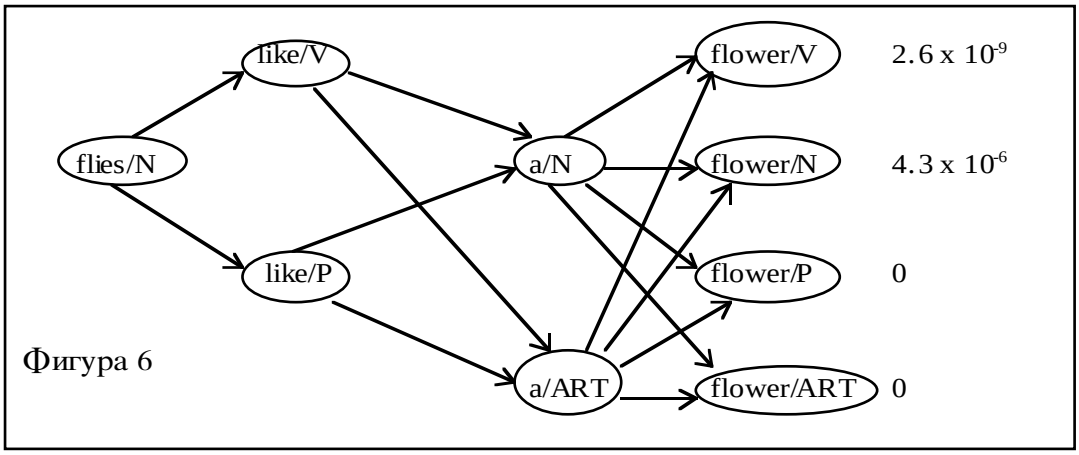
**ЗАБЕЛЕЖКА**: Започнахме с уводен пример как низът КОСА е многозначен като съществително (с 4 примерни форми). Отбележете, че POS-tagger-ът описан по-горе НЯМА да разреши този вид многозначност, понеже той е *обучен* да разпознава само ЧАСТИТЕ НА РЕЧТА (по маркери подобни на изброените в таблица 3). По-нататък ще разгледаме алгоритми за word-sense disambiguation, които третират въпроси свързани с уводния пример (с частичен успех).




Фигура 4. Прилагане на алгоритъма на Витерби, стъпки 1 и 2




Фигура 5.




Фигура 6