Лекция 14: Аксиоматична теория на сложността

Аксиоматична теория на сложността

Възниква през 60-те години на XX век в дисертацията на Мануел Блум. Известна е още като *Машинно независима теория на сложността*. При нея не се предполага конкретен изчислителен модел. Единственото, което се изисква, е да има фиксирана ефективна номерация

$$P_0,\ P_1,\ \ldots$$

на всички програми от този модел. Мерките за сложност, които се разглеждат, се наричат абстрактни мерки за сложност и се въвеждат с двете *аксиоми на Блум*.

За разлика от Аксиоматичната теория на сложността, в *Структурната теория на сложността* се работи в конкретен изчислителен модел — този на машините на Тюринг. Мерките за сложност също са конкретни — това са времевата и пространствената мярка.

5.1 Аксиоми на Блум

Въпреки че в аксиоматичната теория на сложността изчислителният модел няма значение, за удобство ще предполагаме, че продължаваме да работим в нашия модел с МНР, който изучавахме дотук, т.е. навсякъде под "програма" ще разбираме програма за МНР. Принципно ще разглеждаме програми с една входна променлива.

Определение 5.1. Функцията $M\colon\ \mathbb{N}\times\mathbb{N} \rightharpoonup \mathbb{N}$ наричаме *абстрактна мярка за сложност*, ако са изпълнени условията:

(B1) за всяко a и x: $\ !M(a,x) \iff P_a(x)\downarrow$;

(B2) множеството $R = \{(a,x,y) \mid M(a,x) \simeq y\}$ е разрешимо.

Нека $M(a,x)$ е дефинирана. Тогава числото $M(a,x)$ ще тълкуваме като M-ресурса, който се изразходва при работата на P_a върху x. Например $M(a,x)$ може да бъде времето, за което P_a работи върху x или паметта, която се използва при това изчисление. Детайлите ще уточним след малко в раздел 5.2. Сега да обърнем внимание на факта, че говорим за *сложност на програмата* P_a, а не на функцията φ_a. Както ще отбележим по-нататък, някои функции просто нямат най-добра сложност.

От аксиома B1 се вижда, че $Dom(M) = Dom(\Phi_1)$, а аксиома B2 казва, че е разрешима графиката на M, която е точно множеството R. Оттук следва, разбира се, че графиката на M е и полуразрешима, и значи функцията M е изчислима, съгласно теоремата за графиката. Излиза, че функцията "абстрактна мярка" е много особена функция — тя има проста (разрешима) графика и възможно най-сложния домейн — този на универсалната функция. Дали въобще съществуват такива функции? Да, примерите за абстрактни мерки, които ще дадем по-долу, ще са примери за точно такива функции.

Оказва се, че в някои случаи вместо аксиомата B2 е по-удобно да използваме следната аксиома B2′:

(B2′) множеството $R' = \{(a,x,y) \mid M(a,x) \leq y\}$ е разрешимо.

Разрешимостта на R' означава, че има да/не алгоритъм, който за всяка програма P_a, за всеки вход x и за всяко y ни казва дали абстрактният ресурс M, който е в количество y, ще "стигне" за работата на P_a върху x. Оказва се, че условието ресурсът да "стигне" е еквивалентно на условието да "стигне точно", което беше смисълът на аксиома B2. Да се убедим, че е така:

Твърдение 5.1. Аксиомите B2 и B2′ са еквивалентни.

Доказателство. Нека е в сила B2, т.е. $R = \{(a,x,y) \mid M(a,x) \simeq y\}$ е разрешимо множество. Тогава условието за принадлежност към R' на произволна тройка $(a,x,y) \in \mathbb{N}^3$ можем да изразим чрез принадлежност към R по следния начин:

$$(a,x,y) \in R' \iff \exists z_{z \leq y}\ (a,x,z) \in R.$$

Тъй като ограниченият квантор за съществуване запазва разрешимостта (*Твърдение* 4.2), то и R' ще е разрешимо.

Обратно, ако $R' = \{(a,x,y) \mid M(a,x) \leq y\}$ е разрешимо, то за произволни $(a,x,y) \in \mathbb{N}^3$ ще имаме:

$$(a,x,y) \in R \iff (a,x,y) \in R' \ \&\ \underbrace{(y > 0 \implies (a,x,y-1) \notin R')}_{y=0 \ \vee\ (a,x,y-1) \in \overline{R'}}.$$

Следователно множеството R също е разрешимо. □

5.2 Примери за мерки за сложност

При фиксирано a, да означим с M_a функцията, която за за всяко x се дефинира като:

$$M_a(x) \overset{\text{деф}}{\simeq} M(a,x).$$

Функцията M_a ще тълкуваме като *сложност на програмата P_a относно мярката M*. От аксиома B1 имаме, че

$$!M_a(x) \iff !M(a,x) \iff P_a(x)\downarrow,$$

т.е. M_a е дефинирана точно в точките, върху които P_a спира.

Следващите примери показват, че двете естествени понятия за сложностни мерки — памет и време, удовлетворяват аксиомите на Блум.

Твърдение 5.2. Следните функции са мерки за сложност:

1) *Времевата* мярка за сложност

$$T(a,x) \simeq \begin{cases} \text{броя стъпки, за които } P_a \text{ спира върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Доказателство. Аксиомата B1 очевидно е в сила.

Аксиома B2 интуитивно е ясна: за да разберем дали P_a ще спре върху x за t такта, просто я пускаме да работи за тези t такта и гледаме какво се случва.

За строгото доказателство на B2 си даваме сметка, че

$$T(a,x) \simeq t \iff P_a \text{ спира върху } x \text{ за } t \text{ такта} \overset{\text{деф}}{\iff} (a,x,t) \in RHP_1.$$

Но RHP_1 — ограниченият стоп проблем, е разрешим, съгласно *Твърдение 4.22*. Следователно графиката на функцията T е разрешима, с други думи, аксиома B2 е в сила за тази функция. □

2) Времевата мярка T^*, която използва физическото време, а не абстрактните тактове от изчислението на P_a:

$$T^*(a,x) \simeq \begin{cases} \text{времето (примерно в сек), за което } P_a \text{ спира върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Доказателство. Разсъждаваме както при времева мярка T. □

3) *Пространствената* мярка за сложност

$$L(a,x) \simeq \begin{cases} \text{мах стойност на регистър при работата на } P_a \text{ върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Доказателство. Аксиомата B1 отново е ясна. За да проверим B2, да фиксираме произволна програма P_a. Нека тя има q на брой оператора, а m е максималният номер на регистър, който се среща в P_a (което означаваше: всички регистри, които участват в P_a, са сред $X_1, \ldots, X_m$).

Можем да си мислим, че паметта на P_a е $(X_1, \ldots, X_m)$. Тогава всяка конфигурация за тази програма е от вида $(l, x_1, \ldots, x_m)$. Да си дадем сметка, че броят на незаключителните конфигурации, при които съдържанието на регистрите не надвишава y, е точно

$$q(y+1)^m.$$

(За всяко x_i имаме $y+1$ възможни стойности — от 0 до y, а възможните стойностите на адресите l са колкото броя на операторите, т.е. q.) Сега нашата стратегия да разбираме дали $(a, x, y) \in R$ или не, е следната: пускаме програмата P_a да работи върху x точно $t = q(y+1)^m + 1$ такта. Имаме две възможности: за това време P_a да е спряла или да не е спряла.

Ако е спряла, проследяваме изчислението и гледаме съдържанието на регистрите в хода на изчислението. Ако максимумът им е точно y, то $(a, x, y) \in R$, ако не — $(a, x, y) \notin R$.

Ако $P_a(x)$ не е спряла за тези t такта, твърдим, че $(a, x, y) \notin R$. Разглеждаме поотделно двата възможни случая:

1 сл. По време на изчислението в някой от регистрите се е появила стойност, която е по-голяма от y. Тогава по дефиниция $(a, x, y) \notin R$.

2 сл. Няма регистър, чието съдържание да е надхвърлило y на някоя от тези t стъпки. Тъй като за това време P_a не е спряла върху x, то тя не е попадала в заключително състояние. Сега си даваме сметка, че броят на всички незаключителни конфигурации с максимална стойност на регистрите под y, е по-малък от броя на тактовете t, което означава, че P_a е попаднала поне два пъти в една и съща конфигурация. Следователно тя зацикля върху x и значи отново $(a, x, y) \notin R$.

Разбира се, тази процедура е "равномерна" по a, защото броят q на операторите на P_a се изразява чрез a — той е точно $lh(a)$, а като горна граница за паметта на P_a можем да вземем самото a. С други думи, сега алгоритъмът ни започва така: пресмятаме функцията $b(a, x, y) = lh(a)(y+1)^a$. После пускаме P_a да работи върху x точно $b(a, x, y) + 1$ такта. Доказателството, че това е достатъчно, за да можем да определим дали $(a, x, y) \in R$ вече го имаме по-горе. □

Има още много примери за ресурси, които са мерки за сложност. Ето два примера, които остават като задача за допълнителни точки.

Задача 5.1. (Задача за ЕК) Докажете, че следните функции са мерки за сложност:

$$S(a,x) \simeq \begin{cases} \text{броя на изпълнените команди от вида } \mathcal{S}(n) & \\ \text{при работата на } P_a \text{ върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе} \end{cases}$$

$$J(a,x) \simeq \begin{cases} \text{броя на изпълнените команди от вида } J(m,n,q) & \\ \text{при работата на } P_a \text{ върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Функциите, за които дотук показахме, че са абстрактни мерки за сложност, в интуитивен смисъл наистина представляват някакъв критерий за сложност на една програма. Този факт може да се счита като аргумент за това, че аксиомите на Блум са адекватни. Друг аргумент в тази посока е наблюдението, че някои характеристики, които нямаме основание да считаме за критерии за сложност, наистина не удовлетворяват аксиомите на Блум.

По-долу ще дадем няколко примера за такива функции. За да покажем за някои тях, че B2 пропада, ще се възползваме от едно просто наблюдение, което съобразихме в *Задача* 4.2. Според него, ако една функция f има разрешима графика и се мажорира от някоя рекурсивна функция b, т.е. за всяко $\bar{x}$ е изпълнено

$$!f(\bar{x}) \implies f(\bar{x}) \leq b(\bar{x}),$$

то нейната дефиниционна област $Dom(f)$ също е разрешимо множество.

Ние ще прилагаме от този резултат така: за да покажем, че дадена функция M не е мярка, ще е достатъчно да видим, че M се мажорира от някаква рекурсивна функция. Тогава ако допуснем, че M е мярка, то по аксиома B2 тя трябва да има разрешима графика, откъдето по *Задача* 4.2 ще следва, че $Dom(M)$ е разрешимо множество — противоречие с аксиома B1. Следователно функцията M не може да е мярка.

Твърдение 5.3. Следните функции не са мерки за сложност:

1) $M_1(a,x) \stackrel{\text{деф}}{=}$ дължината на работната памет на P_a.

Доказателство. Тази функция е тотална и следователно не удовлетворява още първата аксиома B1. □

2) $$M_2(a,x) \simeq \begin{cases} \text{дължината на работната памет на } P_a & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Доказателство. Тази функция вече удовлетворява B1, обаче B2 пропада. Това е така, защото M_2 се мажорира от рекурсивната функция $b(a,x) \stackrel{\text{деф}}{=} a$. □

3) $$M_3(a,x) \simeq \begin{cases} 0, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Фактът, че тази функция не е мярка показва още веднъж, че няма безплатни изчисления $\ddot{\smile}$.

Доказателство. Функцията M_3 се мажорира от константата 0 и съгласно *Задача* 4.2 не може да бъде абстрактна мярка. □

4) $$M_4(a,x) \simeq \begin{cases} \varphi_a(x), & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе.} \end{cases}$$

Забележка. Това, че функцията M_4 не е мярка за сложност означава, че *резултатът* от работата на P_a върху x не е критерий за сложността на едно изчисление. Може да имаме малък (като стойност) резултат, но той да е получен след немалки изчисления. Например, ако вземем някакво "сложно" разрешимо множество, неговата характеристична функция ще взема само стойности 0 и 1, но ще е трудно да се определи дали стойността е 0 или 1.

Доказателство. Всъщност M_4 е точно универсалната функция Φ_1. Да допуснем, че Φ_1 е мярка за сложност. Вече не можем да ограничим нейните стойности с рекурсивна функция, както беше в примерите дотук. Затова ще се ограничим до една конкретна функция, която има малки стойности, но се пресмята трудно.

Тази функция ще бъде полухарактеристичната функция C_K на множеството K. Нека P_a е програма, която я пресмята, т.е. $\varphi_a = C_K$. Тогава за всяко x ще е изпълнено:

$$x \in K \iff \varphi_a(x) \simeq 0 \iff (a,x,0) \in G_{\Phi_1}.$$

По допускане G_{Φ_1} е разрешимо множество, откъдето следва, че и K трябва да е разрешимо, а то не е — противоречие. □

Ето още един пример за функция, която не е мярка за сложност:

Задача 5.2. (Задача за ЕК) Докажете, че следната функция не е мярка за сложност:

$$Z(a,x) \simeq \begin{cases} \text{броят на изпълнените команди от вида } Z(n) & \\ \text{при работата на } P_a \text{ върху } x, & \text{ако } P_a(x)\downarrow \\ \neg!, & \text{иначе} \end{cases}$$

Упътване. Използвайте, че $Z(n)$ може да се симулира от командата $T(n,m)$ за достатъчно далечно m.

5.3 Теорема за рекурсивна свързаност на мерките

Нека P е произволен едноместен предикат. Ще казваме, че P е в сила *почти навсякъде (п.н.)*, ако съществува число x_0, такова че $P(x)$ е вярно за всяко $x \geq x_0$.

Почти всички твърдения, които ще формулираме оттук нататък, ще са в сила само почти навсякъде. По думите на един от авторите на тази теория, ако се махне това "почти навсякъде", ще остане почти нищо ☹.

Да направим и една уговорка: ако M е мярка за сложност и $\neg!M_a(x)$ (т.е. $M_a(x)$ няма стойност), ще считаме, че тази стойност е ∞. Така е, например, при времевата мярка: ако $\neg!T_a(x)$, това означава, съгласно аксиома B1, че програмата P_a не спира върху x, и значи $T_a(x) = \infty$.

При тази уговорка можем да твърдим следното: ако C и D са мерки, а f е тотална функция, тогава неравенството

$$C_a(x) \leq f(D_a(x))$$

ще е тривиално вярно, ако $\neg!C_a(x)$. Наистина, ако $\neg!C_a(x)$, то по аксиома B1 $\neg!D_a(x)$, което означава, че двете страни на неравенството са ∞, и значи то наистина е вярно. Ще имаме предвид това наблюдение при формулировката на някои от следващите твърдения.

Ще казваме, че функцията $f(x)$ е *повишаваща (scaling factor)*, ако за нея е изпълнено:

1) f е растяща (има се предвид нестрого);

2) $\lim\limits_{x \longrightarrow \infty} f(x) = \infty$.

Повишаващи функции са, да кажем, функциите $2x$, x^2, 2^x, $\lfloor log_2(x) \rfloor$.

Следващото твърдение дава начин, тръгвайки от дадена мярка, да получаваме нови и нови мерки за сложност.

Твърдение 5.4. Нека $C(a,x)$ е мярка за сложност, а f е рекурсивна повишаваща функция. Да дефинираме функцията D като

$$D(a,x) \simeq f(C(a,x))$$

за всяко a и x. Тогава $D(a,x)$ също е мярка за сложност.

Доказателство. Тъй като f е тотална, а C е мярка, аксиомата B1 е в сила и за D.

За да проверим B2, да разпишем условието за принадлежност към графиката на $D = f \circ C$. Имаме

$$D(a,x) \simeq y \iff \exists z\ (\underbrace{C(a,x) \simeq z}_{\text{разрешимо}}\ \&\ \underbrace{f(z) = y}_{\text{разрешимо}}).$$

За да ограничим квантора по z, дефинираме функцията

$$b(y) = \mu z[f(z) > y].$$

Тя е изчислима и тотална (второто е защото f е повишаваща). Сега лесно се съобразява, че за графиката на D е изпълнено:

$$D(a,x) \simeq y \iff \exists z_{z \leq b(y)}\ (\underbrace{C(a,x) \simeq z}_{\text{разрешимо}}\ \&\ \underbrace{f(z) = y}_{\text{разрешимо}})$$

и следователно тя е разрешимо множество. □

От това твърдение следва, че ако тръгнем от функция, която е мярка — примерно от времевата мярка $T(a,x)$, то мерки за сложност ще са и функциите $2.T(a,x)$, $T^2(a,x)$, $2^{T(a,x)}$, $\lfloor log_2(T(a,x)) \rfloor$ и прочее.

Оказва се, че е в сила и резултат, който до известна степен е обратен на горния. Той показва, че всеки две мерки за сложност са свързани, в смисъл, че ако една програма P_a има малка сложност по отношение на едната мярка, то тя ще има малка сложност и по отношение на другата мярка.

Теорема 5.1. (Теорема за рекурсивната свързаност на мерките) Нека C и D са мерки за сложност. Съществува рекурсивна функция $f(x,y)$, строго растяща по y, такава че:

$$C_a(x) \leq f(x, D_a(x)) \quad \text{за всяко } x \geq a;$$

$$D_a(x) \leq f(x, C_a(x)) \quad \text{за всяко } x \geq a.$$

Забележка. От това, че f е растяща по y следва, че ако $D_a(x)$ е малко число, то $f(x, D_a(x))$ не може да е голямо, откъдето и $C_a(x)$, което се ограничава от $f(x, D_a(x))$, също не може да е голямо. С други думи, ако по отношение на една мярка програмата P_a е проста, то по отношение на коя да е друга мярка тя не може да е сложна.

Доказателство. Избираме си следната функция

На лекции взехме теоремата без доказателство

$$h(a,x,y) = \begin{cases} max(C(a,x), D(a,x)), & \text{ако } C_a(x) \simeq y\ \vee\ D_a(x) \simeq y \\ 0, & \text{иначе.} \end{cases}$$

Тъй като графиките на C и D са разрешими, то h е изчислима. Тя очевидно е и тотална, тъй че общо h е рекурсивна функция.

Сега да вземем f да е следната функция:

$$f(x,y) \;=\; y + \max_{a\le x}(\max_{z\le y} h(a,x,z)). \tag{5.1}$$

Лесно се вижда, че f е рекурсивна и строго растяща по y. Да се убедим, че тя има свойството от теоремата. За целта да фиксираме a, вземем произволно $x \ge a$. Ако $\neg!C_a(x)$, и двете неравенства са тривиално верни. Да приемем сега, че $!C_a(x)$. Тогава $!D_a(x)$ и

$$\begin{aligned} f(x, D_a(x)) &\overset{\text{деф}}{=} D_a(x) + \max_{i\le x}(\max_{z\le D_a(x)} h(i,x,z)) \\ &\ge \max_{i\in\{0,\ldots,a,\ldots,x\}}(\max_{z\le D_a(x)} h(i,x,z)) \;\ge\; \max_{z\le D_a(x)} h(a,x,z) \\ &\ge\; h(a,x,D_a(x)) \overset{\text{деф}}{=} max(C_a(x), D_a(x)) \;\ge\; C_a(x). \end{aligned}$$

Аналогично се доказва и другото неравенство. □

Дали може горната теорема да бъде във вид, по-близък до предишното *Твърдение* 5.4? Може, но при условие, че $C_a(x) \ge x$ и $D_a(x) \ge x$.

Това условие наистина е необходимо. Лесно се вижда, че ако в качеството на C и D вземем времевата мярка T и пространствената мярка L, то за програмата $P_a\colon X_1 := X_1 + 1$ ще имаме за всяко x

$$T_a(x) = 1 \quad \text{и} \quad L_a(x) = x + 1.$$

Следователно не е възможно за всяко x

$$L_a(x) \le r(T_a(x)) = r(1) = const.$$

Следствие 5.1. Нека $C_a(x) \ge x$ и $D_a(x) \ge x$ за всяко x. Тогава съществува строго растяща рекурсивна функция r, такава че:

$$C_a(x) \le r(D_a(x)) \quad \text{за всяко } x \ge a;$$

$$D_a(x) \le r(C_a(x)) \quad \text{за всяко } x \ge a.$$

Доказателство. Тръгваме от функцията f, която дефинирахме по-горе с равенството (5.1) и чрез нея конструираме следната функция r:

$$r(y) \;=\; \max_{z\le y} f(z,y).$$

Ясно е, че r е рекурсивна и строго растяща. Да се убедим, че тя върши работа. Отново фиксираме a, вземаме произволно $x \ge a$ и приемаме, че $!C_a(x)$ (което значи и $!D_a(x)$). Тогава, тъй като $D_a(x) \ge x$, ще имаме:

$$r(D_a(x)) \overset{\text{деф}}{=} \max_{z\in\{0,\ldots,x,\ldots,D_a(x)\}} f(z, D_a(x)) \ge f(x, D_a(x)) \ge C_a(x).$$

Аналогично разсъждаваме и за второто неравенство. □

5.4 Класове на сложност. Теорема за пропастта

Нека M е мярка за сложност, а b е рекурсивна функция. Да напомним, че M_a се мажорира от b (запис: $M_a \leq b$), когато

$$\forall x(!M_a(x) \implies M_a(x) \leq b(x)).$$

Ще пишем

$$M_a \leq b \text{ п.н.},$$

когато импликацията $!M_a(x) \implies M_a(x) \leq b(x)$ е в сила за всички x, с изключение на краен брой, т. е. за всички x над някакво x_0.

Ако M е мярка за сложност, а b е рекурсивна функция, с C_b^M ще означаваме следния клас от едноместни рекурсивни функции:

$$C_b^M = \{f \mid f \text{ е тотална } \& \ \exists a\ (\varphi_a = f \ \& \ M_a \leq b \text{ п.н.})\ \}.$$

С други думи, в класа C_b^M са всички едноместни рекурсивни функции, които могат да се изчислят с M-сложност под b.

Класът C_b^M ще наричаме *сложностен клас* (или *клас на сложност*) относно мярката M.

Забележете, че сложностният клас C_b^M съдържа само рекурсивни функции. Какво ще стане, ако в него включим *всички* изчислими едноместни функции? Ето какво:

Задача 5.3. Да означим

$$\hat{C}_b^M = \{f \mid \exists a\ (\varphi_a = f \ \& \ M_a \leq b \text{ п.н.})\ \}.$$

Докажете, че всяка функция от този клас е с разрешимо дефиниционното множество (и следователно тя е потенциално рекурсивна функция).

Доказателство. Нека $f \in \hat{C}_b^M$ и нека програмата P_a пресмята f с M-сложност, която е под b. С други думи, $\varphi_a = f$ и$M_a \leq b$ п.н. Нека още x_0 е такова, че за всяко $x \geq x_0$ е изпълнено $M_a(x) \leq b(x)$. Ще покажем, че $Dom(f)$ е разрешимо множество. Наистина, да изберем произволно $x \geq x_0$. Тогава

$$x \in Dom(f) \iff !\varphi_a(x) \overset{\text{B1}}{\iff} !M_a(x) \iff \exists z M_a(x) \simeq z \iff \exists z_{\leq b(x)} M_a(x) \simeq z.$$

Съгласно аксиома B2, множеството $\{(x,z) \mid M_a(x) \simeq z\ \}$ е разрешимо, Следователно $\{x \mid x \geq x_0 \ \& \ x \in Dom(f))$ е разрешимо, откъдето и цялото множество $Dom(f)$ е разрешимо. $\square$

Любопитно е да се отбележи, че класът на всички едноместни примитивно рекурсивни функции е сложностен клас относно времевата и пространствената мярка, т.е. съществуват рекурсивни функции b_T и b_L, такива че

$$\mathcal{PR}_1 = C^T_{b_T} \quad \text{и} \quad \mathcal{PR}_1 = C^L_{b_L}.$$

Нека b и b' са произволни рекурсивни функции. Интуицията ни подсказва, че ако b' мажорира b, при това многократно (примерно ако $b'(x) = 2^{b(x)}$), то би трябвало класът $C^M_{b'}$ да включва строго класа C^M_b. Оказва се обаче, че ако функциите b и b' се подберат по специален начин, това няма да е така. Този резултат ще е следствие от следващата теорема, доказана от Бородин.

Теорема 5.2. (Теорема за пропастта/gap theorem) Нека M е сложностна мярка, а $g(x,y)$ е рекурсивна функция. Съществува рекурсивна функция $b(x)$, такава че за всяко a и всяко $x \geq a$:

$$M_a(x) \leq g(x, b(x)) \implies M_a(x) \leq b(x). \tag{5.2}$$

Преди да сме се заели с доказателството на теоремата, да отбележим, че като директно следствие от нея получаваме, че има функции b, за които например класовете C^M_b и $C^M_{2^b}$ съвпадат. Наистина, нека $b(x)$ е функцията от теоремата за пропастта, приложена за $g(x,y) = 2^y$. От тази теорема ще имаме, че за всяко a и всяко $x \geq a$:

$$M_a(x) \leq 2^{b(x)} \implies M_a(x) \leq b(x).$$

Ясно е, че $C^M_b \subseteq C^M_{2^b}$. За да видим обратното, да вземем произволна f от класа $C^M_{2^b}$. За нея ще съществува програма P_a, която я пресмята, такава че $M_a \leq 2^b$ п.н. От (5.2) получаваме, че всъщност $M_a \leq b$ п.н., което означава, че $f \in C^M_b$.

Сега да пристъпим към доказателството на теоремата.

Доказателство. Да разгледаме множеството

И тази теорема е без доказателство :).

$$A = \{ (x,y) \mid \forall a \leq x \ (M_a(x) \leq y \ \vee \ M_a(x) > g(x,y)) \}.$$

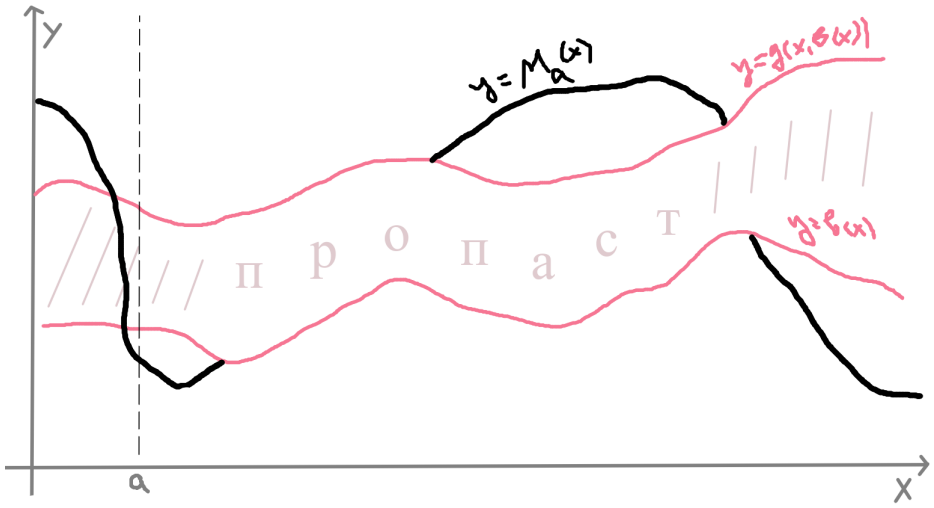
Условието $M_a(x) \leq y$ е разрешимо по аксиома B2, а другото условие $M_a(x) > g(x,y)$ можем да препишем така:

$$M_a(x) > g(x,y) \iff \forall z \leq g(x,y) \ \neg M_a(x) \simeq z.$$

Следователно множеството A е разрешимо. Нека

$$b(x) \overset{\text{деф}}{=} \mu y[\chi_A(x,y) = 0].$$

Тази функция е изчислима. За да покажем, че е тотална, трябва да видим, че за всяко x съществува $y : (x,y) \in A$.



Наистина, да фиксираме едно естествено число x и да разгледаме множеството $B = \{\ M_a(x) \mid !M_a(x)\ \&\ a \leq x\ \}$, с други думи,

$$B\ =\ \{M_0(x), \ldots, M_x(x)\},$$

като приемаме, че ако $\neg!M_a(x)$, то не добавяме стойност в B. Множеството B е крайно и следователно има най-голям елемент y (ако $B = \emptyset$, нека $y \stackrel{\text{деф}}{=} 0$). Ще покажем, че $(x, y) \in A$, откъдето ще следва, че $!b(x)$. И понеже x е произволно, то значи функцията b ще е тотална и следователно — рекурсивна.

Наистина, да вземем някакво $a \leq x$. Ако $!M_a(x)$, то $!M_a(x) \leq y$, защото y беше максималният елемент на B. Следователно $(x, y) \in A$. Ако $\neg!M_a(x)$, то $M_a(x)$ е по-голямо от всяко нещо и в частност, $M_a(x) > g(x, y)$. Значи отново $(x, y) \in A$.

Сега пристъпваме към доказателството на импликацията (5.2) от теоремата. За целта фиксираме някакво $a \in \mathbb{N}$ и нека $x \geq a$ е произволно. Да приемем, че предпоставката на (5.2) е налице, т.е. $M_a(x) \leq g(x, b(x))$.

От друга страна, по определение $(x, b(x)) \in A$, освен това $a \leq x$, и значи е вярно, че

$$M_a(x) \leq y\ \ \vee\ \ M_a(x) > g(x, y).$$

Но второто условие $M_a(x) > g(x, y)$ не е изпълнено, и остава да е вярно първото — $M_a(x) \leq y$. $\square$

5.5 Теорема за ускорението

Теоремата за пропастта, която доказахме, е единият от двата т. нар. "странни резултати" в абстрактната теория на сложността. Вторият резултат (който всъщност е по-странният), е *теоремата за ускорението*, доказана най-напред от Блум.

При фиксирана мярка M е логично да очакваме, че за всяка изчислима функция съществува най-добра (оптимална) по отношение на M програма, която я пресмята. Това, обаче, не е така. Оказва се, че съществуват функции (които могат дори да са 0-1-функции), такива че *всяка* програма, която ги пресмята, може да бъде оптимизирана (или ускорена), при това колкото си искаме много. Ето и точната формулировка:

Теорема 5.3. (Теорема за ускорението/speedup theorem) Нека M е мярка за сложност, а $r(x,y)$ е произволна рекурсивна функция. Съществува рекурсивна функция f със свойството: за всяка програма P_a, пресмятаща f, съществува програма P_b, която също пресмята f, такава че

$$r(x, M_b(x)) \leq M_a(x) \quad \text{почти навсякъде.}$$

Ще пропуснем доказателството на тази теорема, тъй като то изисква една отделна лекция ☺. Вместо това ще си направим няколко експеримента. Да вземем например $r(x,y) = 100y$. Тогава ще излезе, че

$$100M_b(x) \leq M_a(x) \quad \text{п.н.,}$$

или все едно

$$M_b(x) \leq \frac{M_a(x)}{100} \quad \text{п.н.,}$$

т.е. постигнали сме стократно намаление на сложността.

Ако вземем $r(x,y) = 2^y$, ще имаме

$$2^{M_b(x)} \leq M_a(x) \quad \text{п.н.,}$$

което означава, че сме получили умопомрачителното логаритмично намаление на сложността на P_a:

$$M_b(x) \leq log_2(M_a(x)) \quad \text{п.н.}$$