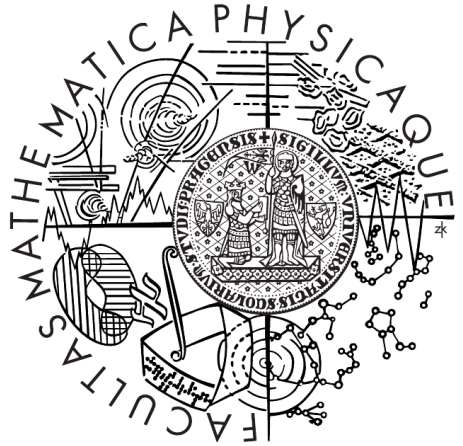
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Bc. Petra Kuřinová

Polynomiální algoritmus pro binární PCP

Katedra algebry

Vedoucí diplomové práce:	doc. Mgr. Štěpán Holub, Ph.D.
Studijní program:	Matematika
Studijní obor:	Matematické metody informační bezpečnosti

Praha 2012

Na tomto místě bych velmi ráda poděkovala všem, díky nimž jsem tuto práci dokončila. Hlavní dík patří doc. Mgr. Štěpánu Holubovi, Ph.D., který se mi ochotně věnoval, trpělivě odpovídal na mé dotazy, podnětně připomínkoval každý můj text a se zájmem se mnou diskutoval o nových problémech. Aplikaci bych nedokázala naprogramovat nebýt RNDr. Martina Pergela, Ph.D., který mě naučil základy objektově orientovaného programování a byl ochoten zodpovídat jakékoliv otázky z tohoto oboru. Samozřejmě bych chtěla také zmínit osoby z mého blízkého okolí, jimž patří poděkování za podporu a porozumění, které během psaní práce projevovali.

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Praze dne 7. 12. 2012

Název práce: Polynomiální algoritmus pro binární PCP

Autor: Petra Kuřinová

Katedra: Katedra algebry

Vedoucí diplomové práce: doc. Mgr. Štěpán Holub, Ph.D., Katedra algebry

Abstrakt: Postův korespondenční problém, zavedený Emilem Postem v roce 1946 ([8]), je důležitým příkladem obecně nerozhodnutelného problému. Díky tomu figuruje v důkazech některých výsledků z teorie formálních jazyků, teorie matic a dalších odvětví. Rozhodnutelnost binárního Postova korespondenčího problému dokázali Ehrenfeucht, Karhumäki a Rozenberg v 80. letech ([2]) a v roce 2002 Halava, Harju a Hirvensalo důkaz dokončili v [3]. O osm let později bylo v článku [4] ověřeno, že řešení lze nalézt dokonce v polynomiálním čase.
Tato diplomová práce má za hlavní cíl podrobně popsat tento polynomiální algoritmus a implementovat jej v rámci webové aplikace. Práce mimo jiné seznamuje se základy kombinatoriky na slovech a různými poznatky o PCP a také předkládá některé zajímavé instance PCP.

Klíčová slova: Postův korespondenční problém, zobecněný Postův korespondenční problém, binární PCP, polynomiální algoritmy na slovech, následníci homomorfismů

Title: A polynomial algorithm for the binary PCP

Author: Petra Kuřinová

Department: Department of algebra

Supervisor: doc. Mgr. Štěpán Holub, Ph.D., Department of algebra

Abstract: The Post correspondence problem, introduced in 1946 by Emil Post ([8]), is an important example of undecidable problem. Therefore PCP figures in proofs of some results in theory of formal languages, matrix theory and other. The decidability of the Post correspondence problem proved Ehrenfeucht, Karhumäki and Rozenberg in the 1980s ([2]) and Halava, Harju and Hirvensalo in 2002 ended the proof - ([3]). Eight years later was in [4] verified that the solution can be found even in polynomial time.
The main goal of this diploma thesis is to describe this algorithm in detail and to implement it in a web application. The thesis also introduces basics of combinatorics on words and some facts about PCP and produces some interesting examples of instances of PCP.

Keywords: Post correspondence problem, generalized Post correspondence problem, binary PCP, polynomial algorithms on words, successors of morphisms

Obsah

Úvod

Postův korespondenční problém (PCP) patří mezi rozhodovací problémy z oblasti kombinatoriky na slovech. Instanci binární verze tohoto problému tvoří dvojice homomorfismů (g, h) nad dvouprvkovou abecedou, tedy $g, h : \{a, b\}^* \to \{a, b\}^*$. Tato instance má řešení, právě když existuje slovo $w \in \{a, b\}^+$, které má shodné obrazy obou homomorfismů, tedy $g(w) = h(w)$. Podstatou PCP je tedy rozhodnout, zda vstupní instance má či nemá řešení.

Příklad 1. Mějme vstupní instanci (g, h), kde homomorfismy g a h jsou zadané hodnotami:

$$\begin{aligned} g(a) &= aba, \quad & g(b) &= bb, \\ h(a) &= a, \quad & h(b) &= babb. \end{aligned}$$

Pokusme se nejprve konstruovat řešení po písmenech. Na začátku vezmeme prázdné slovo w a budeme jej prodlužovat tak, aby stále $g(w)$ bylo *prefixem* (začátkem) $h(w)$ nebo naopak. Pokračujeme, dokud w nebude řešením dané instance a dokud je po prodloužení splněná podmínka *porovnatelnosti* obrazů. Slovo w budeme nazývat *předřešení*. Na začátku nevíme, kterým písmenem by mělo řešení začít, vyzkoušejme tedy postupně obě možnosti.
Pokud začneme s písmenem b, je $g(b) = bb$, $h(b) = babb$. Tato slova nejsou porovnatelná - na druhém místě se obrazy liší, takže žádné řešení této instance nezačíná písmenem b. První větev tedy vedla k rychlému konci.
Zkusme začít druhým písmenem, písmenem a. Slova

$$g(a) = aba,$$

$$h(a) = a$$

mají shodný začátek (v našem případě je $h(a)$ prefixem slova $g(a)$), nedochází zde k rozporu jako v prvním případě. Nyní by se mohla vyzkoušet obě možná prodloužení, tedy předřešení aa a ab, ale není to nutné. Všimněme si, že slovo $g(a)$ je delší než $h(a)$, vzniká *převis* z, tedy slovo splňující $g(a) = h(a)z$. Ten nám určuje jediné možné pokračování našeho předřešení, obraz homomorfismu h přidaného písmene totiž musí být porovnatelný s tímto převisem. První písmeno převisu $z = ba$ je b, přitom první písmeno slova $h(a)$ je a, takže písmenem a jistě nelze na předřešení navázat. Proto máme předřešení ab, které je již řešením této instance.

Základní myšlenka při hledání řešení binárního PCP je tedy jasná - podle převisů prodlužovat předřešení, dokud nedojdeme ke sporu nebo k řešení. Samozřejmě zde můžeme narazit na několik nepříjemností, které budeme muset na následujících stránkách podrobně rozebrat. První z nich je, že ukončovací podmínka při prodlužování předřešení nemusí nikdy nastat. Je tedy třeba nalézt nějaké další pravidlo, které nekonečnému přidávání písmen zabrání.
Dále vidíme, že pokud bude například homomorfismus g *periodický*, tedy pokud $g(a) = u^k$, $g(b) = u^l$ pro nějaké neprázdné slovo u a $k, l \geq 0$, nepoznáme podle převisu, které slovo by mělo na předřešení navázat. Pro periodické homomorfismy budeme tedy muset postup mírně upravit.

Navíc chceme nalézt algoritmus, který bude hledat řešení v polynomiálním čase. Jestliže při sestrojování předřešení nastane situace, ve které neumíme jednoznačně rozhodnout o jeho jednoznačném prodloužení, musíme vyzkoušet obě možnosti pokračování. Pokud však takových případů bude více (tj. nekonstantně mnoho), bude počet kroků algoritmu exponenciální. Je tedy zapotřebí detekovat instance, u kterých by mohlo k více větvením dojít a nalézt pro ně vhodnější postup. Alespoň zde naznačíme, jakým způsobem toho docílíme. Tato tvrzení uvádíme zatím bez důkazů, později v práci samozřejmě dokázaná budou.

Tvrzení. *Mějme $I = (g, h)$ instanci PCP, w její řešení. Pak lze w rozložit na*

$$w = u_1 \dots u_k = v_1 \dots v_k,$$

kde dvojice (u_i, v_i) jsou pro každé $i = 1, 2, \dots, k$ takové, že

1. *$g(u_i) = h(v_i)$,*
2. *pro žádnou dvojici (u'_i, v'_i) takovou, že u'_i je prefix u_i, v'_i je prefix v_i, neplatí $g(u'_i) = h(v'_i)$.*

Dvojici slov (e, f) splňující podmínky 1. a 2. nazveme *blok instance I*. Navíc lze dokázat, že pro každou instanci existují nejvýš dva bloky. Přitom ještě platí:

Lemma. *Jsou-li (e, f) a (e', f') dva bloky instance I, bude první písmeno slova e různé od prvního písmene slova e' a také se na prvních místech liší písmena slov f a f'.*

Příklad 2. Představme si na vstupu instanci (g, h) zadanou takto:

$$\begin{aligned} g(a) &= a, \quad & g(b) &= bb, \\ h(a) &= abb, \quad & h(b) &= b. \end{aligned}$$

Zkusme nalézt bloky této instance. Začněme sestrojovat první blok (e, f) od písmene a, tj. máme počátek bloku (a, ϵ), kde ϵ značí prázdné slovo. Podle převisu jeho obrazů zjistíme, že druhé slovo musí začínat písmenem a. Potom nás převis slov $g(a)$, $h(a)$ zavede k pokračování b u prvního slova, a blok (ab, a) je nalezen. Zkusíme-li začít písmenem b při konstruování bloku (e', f'), podle převisu je prvním písmenem druhého slova písmeno b. Dále nám převis obrazů těchto slov napoví, že je třeba ke druhému slovu přidat písmeno b a vznikne tak blok (b, bb). Podle uvedeného Tvrzení tedy každé řešení musí být nějakým uspořádáním bloků. Nyní si můžeme všimnout, že výsledné řešení $w = abb$ je zřetězením prvních, resp. druhých, slov z obou bloků - v tomto případě tedy $w = ee' = ff'$.
Ovšem problém jak uspořádat bloky, abychom dostali řešení, je vlastně problémem PCP. Mohli bychom tedy definovat instanci (g_1, h_1) s obrazy homomorfismů danými právě těmito bloky, v našem případě vytvoříme instanci (g_1, h_1) takto:

$$\begin{aligned} g_1(a) &= ab, \quad & g_1(b) &= b, \\ h_1(a) &= a, \quad & h_1(b) &= bb. \end{aligned}$$

Řešení této instance je slovo ab, to tedy říká, že v řešení původní instance máme za blok začínající písmenem a umístit blok od písmene b.

Dvojici homomorfismů (g_1, h_1), jejíž obrazy jsou bloky původní instance (jako tomu bylo v předchozím příkladě), nazveme *následníkem* původní instance. Takto lze sestrojit posloupnost následníků. Lze dokázat, že řešení následníka je ve většině případů jadnodušší než řešení původní instance, následník má totiž obrazy většinou kratší a může vznikat menší počet převisů.

Problémy v tomto postupu jsou dva:

- u některých instancí se řešení pro následníka nezjednoduší, některé instance nemají oba bloky, tedy ani následníka,
- u instancí, které nejsou tzv. *markované*, tj. obrazy některého z homomorfismů začínají týmž písmenem, jsou bloky složitější a přechod k následníkovi bude znamenat přechod k obecnějšímu problému, GPCP.

Lemma výše nám vlastně říká, že následníci jsou markovaní. Pokud tedy začneme s markovaným homomorfismem, k přechodu ke GPCP vůbec nedojde. Řešení těchto potíží však již přenecháme samotné práci.

1. Základní pojmy a myšlenky

Před samotným řešením problému je samozřejmě nutné ujasnit si definice některých obecných pojmů, na kterých budeme stavět, aby nedošlo k nedorozumění. Některé z nich se týkají výhradně Postova korespondenčního problému, budeme ovšem používat i několik obecnějších definic především z oblasti kombinatoriky na slovech.

1.1 Kombinatorika na slovech

Většina z následujících definic a tvrzení je k nalezení téměř ve všech knihách o kombinatorice na slovech, například v 6 a 7.

Definice. *Abeceda* A je neprázdná konečná množina symbolů, které budeme nazývat *písmena*. Konečnou posloupnost písmen z abecedy A nazveme *slovo (nad abecedou A)*.

Slova nad abecedou A s operací zřetězení (někdy značenou $||$, symbol je ale zvykem vynechávat) a prázdným slovem ϵ jako jednotkou tvoří monoid, který značíme A^*. Jsou totiž splněné všechny podmínky z definice monoidu:

- uzavřenost: pro každé $a, b \in A^*$ je jistě $ab \in A^*$,
- asociativita: $(ab)c = a(bc)$ pro každá tři slova a, b, c nad abecedou A a
- prázdné slovo: $\epsilon a = a\epsilon = a$ pro každé $a \in A^*$.

Pologrupu všech neprázdných slov nad A značíme A^+ a délku slova u (neboli počet písmen, ze kterých sestává) budeme zapisovat jako $|u|$.

Definice. Slovo $u \in A^*$ budeme nazývat *prefix slova* $v \in A^*$, jestliže existuje $w \in A^*$ takové, že $v = uw$. To znamená, že slovo v začíná slovem u. Tuto relaci pak označujeme $u \leq v$.
První písmeno neprázdného slova v značíme $\text{pref}_1(v)$.
Řekneme, že slova u a v jsou *prefixově porovnatelná*, $u \bowtie v$, je-li jedno prefixem druhého, tedy jestliže $u \leq v$ nebo $v \leq u$. Nejdelší společný prefix slov u a v značíme $u \wedge v$.

Obdobné pojmy definujeme i pro slova, která jsou porovnatelná „odzadu“.

Definice. Slovo $u \in A^*$ je *sufix slova* $v \in A^*$, pokud existuje $w \in A^*$ takové, že $v = wu$, neboli v končí slovem u. Relaci „u je sufixem v“ značíme $u \leq_s v$.
Slova u a v nazveme *sufixově porovnatelná,* pokud jedno z nich je sufixem druhého.
Nejdelší společný sufix slov u, v značíme $u \wedge_s v$.
Množinu všech sufixů slova u značíme $\text{suff}(u)$, tedy

$$\text{suff}(u) = \{x \in \{a, b\}^* | \exists v \in \{a, b\}^* : u = vx\}.$$

Neprázdný prefix, resp. sufix, u slova v nazveme *vlastní*, je-li $u \neq v$.

Jak bylo napsáno již v úvodní kapitole, budeme ve velké míře pracovat s převisy. Máme-li dvě porovnatelná slova $u \leq v$, chceme se jednoduše dostat k jejich převisu, tj. ke slovu w takovému, že $uw = v$. Vidíme, že stačí od začátku slova v „odebrat“ slovo u. Tato operace je inverzní k operaci zřetězení, označíme ji $^{-1}$. Převis w tedy dostaneme jako $w = u^{-1}v$. Touto operací lze monoid A^* rozšířit do volné grupy generované A. Je-li například $u = vw$, platí $v = uw^{-1}$, $w = v^{-1}u$ a také $v^{-1} = wu^{-1}$. Zápisem $w = u^k$ pro nějaké slovo u a hodnotu $k \in \mathbb{N}_0$ budeme označovat skutečnost, že slovo w sestává z k slov u zřetězených za sebou. Přitom pro $k = 0$ bude $w = \epsilon$.
Na monoidu $A^* \times A^*$ s operací $\cdot$ definovanou $(u, v) \cdot (u', v') = (uu', vv')$ můžeme také zavést prefixové a sufixové uspořádání. Definujeme, že (u, v) je *prefixem* (u', v'), značíme $(u, v) \leq (u', v')$, právě když $u \leq u'$ a zároveň $v \leq v'$. Druhé uspořádání definujeme analogicky: (u, v) je *sufixem* (u', v'), značíme $(u, v) \leq_s (u', v')$, právě když $u \leq_s u'$ a $v \leq_s v'$. Také definujeme $(u, v) = (u', v')$, právě když $u = u'$ a $v = v'$.
O dvojici (u, v) tedy řekneme, že je to *prefixově minimální* dvojice s vlastností V, právě když pro každou dvojici slov (u', v') s vlastností V platí:

$$(u', v') \leq (u, v) \implies (u', v') = (u, v).$$

Sufixově minimální dvojici slov můžeme definovat analogicky, tedy pouze záměnou nerovnosti za $\leq_s$.
Z algebry víme, že lze mezi monoidy definovat lineární zobrazení. Jejich prostřednictvím definujeme problém PCP pro naše účely, takže homomorfismy hrají hlavní roli od této chvíle až do konce poslední kapitoly. Následující definici je třeba si důkladně osvojit.

Definice. Nechť A a B jsou dvě abecedy. Zobrazení $g : A^* \to B^*$ nazveme *homomorfismus*, platí-li pro všechna $u, v \in A^*$ vztah $g(uv) = g(u)g(v)$.
Jestliže $g(u) \neq \epsilon$ pro všechna $u \in A^*$, řekneme, že g je *nenulující*.
Homomorfismus $g : A^* \to B^*$ nazveme *periodický*, existuje-li slovo $w \in B^*$ takové, že pro všechna $u \in A^*$ je $g(u)$ mocninou w, tedy $g(u) = ww \dots w$.

Chceme-li definovat nějaký konkrétní homomorfismus g, nejjednodušší cestou je určení jeho obrazů na jednotlivých písmenech abecedy A, která tvoří bázi monoidu A^*. Potom obraz libovolného slova u nad abecedou A získáme zřetězením obrazů jednotlivých písmen slova u, tedy $g(u) = g(l_1) \cdots g(l_d)$, kde $u = l_1 \cdots l_d$ a $l_i \in A$ pro každé $i \in \{1, \cdots, d\}$.
Příkladem periodického homomorfismu nad dvouprvkovou abecedou $A = \{a, b\}$ je homomorfismus $g : A^* \to A^*$ daný hodnotami:

$$g(a) = ababab, \quad g(b) = abab.$$

Můžeme psát $g(a) = (ab)^3$ a $g(b) = (ab)^2$, slovo ab tvoří tzv. *kořen* tohoto periodického homomorfismu, jak uvádí následující definice.

Definice. Mějme w neprázdné slovo. Nejkratší slovo u splňující $w = u^k$ pro nějaké přirozené k nazveme *kořen* w. Máme-li periodický homomorfismus g daný rovnostmi $g(a) = u^k$ a $g(b) = u^l$ a u je minimální takové, říkáme o u, že je *kořen homomorfismu* g.
Každé slovo, které není mocninou žádného svého vlastního podslova, nazveme *primitivní*. Speciálně platí, že kořen libovolného slova je primitivní.

Jakékoliv slovo také můžeme „číst odzadu“, tedy obrátit jeho pořadí písmen. Přestože se to může jevit jen jako drobná hříčka, později nám tento zpětný pohled může velmi pomoci. Zavedeme proto tuto operaci přesně:

Definice. Je-li $u = l_1 l_2 \dots l_d$ slovo, kde $l_1, l_2, \dots, l_d$ jsou písmena, pak definujeme $\overline{u}$ *zrcadlový obraz slova* u jako slovo $\overline{u} = l_d l_{d-1} \dots l_1$.
Pro každý homomorfismus $g : A^* \to B^*$ definujeme *zrcadlový obraz homomorfismu* g jako homomorfismus $\overline{g} : A^* \to B^*$ definovaný vztahem $\overline{g}(x) = \overline{g(x)}$ pro všechna písmena $x \in A$.

Všimněme si, že obecně pro slovo $u \in A^*$ se $\overline{g}(u)$ nerovná ani $g(\overline{u})$ ani $\overline{g(u)}$, ale platí $\overline{g}(\overline{u}) = \overline{g(u)}$. Mějme homomorfismus g a slovo $g(ba)$. Pro zrcadlový obraz tohoto slova platí

$$\overline{g(ba)} = \overline{g(b)g(a)}.$$

Takové slovo je podle definice zrcadlového obrazu právě slovo $\overline{g(a)}\ \overline{g(b)}$, což při definici $\overline{g}$ znamená, že

$$\overline{g(ba)} = \overline{g}(a)\overline{g}(b).$$

Proto platí $\overline{g(ba)} = \overline{g}(ab)$.
Především při dokazování některých tvrzení o periodických instancích PCP nás bude zajímat, zda lze kořeny periodických homomorfismů řetězit do nekonečna za sebe tak, aby slova byla stále porovnatelná. Toho dosáhneme pouze pokud je jedno slovo kořenem druhého nebo pokud je kořen jednoho slova cyklickým posunem písmen kořene druhého z nich.

Definice. Slova u, v jsou *konjugovaná,* jestliže existují slova x, y taková, že $u = xy$ a $v = yx$. Dá se říci, že písmena slova u jsou ve slově v pouze cyklicky posunuta.

Lemma 1.1. *Následující tvrzení jsou ekvivalentní:*

1. *u a v jsou konjugovaná,*
2. *existuje slovo z, pro které je $uz = zv$,*
3. *kořen u a kořen v jsou konjugovaná slova.*

Důkaz. Důkaz lze najít např. v [1]. □

Lemma 1.2. *Mají-li slova u^∞ a v^∞ společný faktor délky $|uv|$, jsou jejich kořeny konjugovaná slova.*

Důkaz. Důkaz lze najít např. v [1]. □

Tvrzení 1.3. *Mějme dvě primitivní slova u a v a dvě slova p a q splňující $q \leq p$. Nechť platí $pu^\infty = qv^\infty$. Potom $|u| = |v|$ a*

$$q^{-1}pu = vq^{-1}p. \tag{1.1}$$

Navíc je-li splněna rovnice (1.1)*, platí $pu^\infty = qv^\infty$.*

Důkaz. Z Lemmatu 1.2 plyne, že u a v mají konjugované kořeny. Protože jsou primitivní, jsou konjugovaná přímo slova u a v, speciálně mají stejnou délku.
Je-li $pu^\infty = qv^\infty$, $q \leq p$, platí i $q^{-1}pu^\infty = v^\infty$. To znamená, že $q^{-1}pu = vx$ pro nějaké slovo x, $|x| = |q^{-1}p|$. Navíc protože $q^{-1}pu^2 = v^2x$, máme $vxu = v^2x$, takže $x \bowtie v$. Víme ale, že $q^{-1}p \bowtie v$, proto $x = q^{-1}p$ a je splněna rovnice (1.1).
Nechť naopak platí rovnice (1.1). To znamená, že $pu = qvq^{-1}p$, a tedy

$$pu^n = qvq^{-1}p(p^{-1}qvq^{-1}p)^{n-1} = qv^nq^{-1}p.$$

Pro n jdoucí k nekonečnu tedy máme $pu^\infty = qv^\infty$. □

Lemma 1.4. *Neprázdné slovo x je primitivní, právě když pro každé slovo y platí:*

$$yx \leq x^2 \implies (y = \epsilon) \text{ nebo } (y = x).$$

Důkaz. Lemma i s důkazem lze najít např. v [1] pod názvem Primitivity Lemma. □

Definice. Řekneme, že slova $w_1, w_2, \ldots, w_n$ *mají netriviální vztah*, jestliže existují dvě různé posloupnosti indexů z množiny $\{1, 2, \ldots, n\}$, že platí rovnost

$$w_{i_1}w_{i_2}\ldots w_{i_k} = w_{j_1}w_{j_2}\ldots w_{j_m}.$$

Lemma 1.5. *Pokud slova u, v mají netriviální vztah, jsou to mocniny jednoho slova.*

Důkaz. Důkaz tohoto Lemmatu je celkem obtížný, podrobně je uveden například v knize [7]. □

1.2 Definice a tvrzení související s PCP

Emil Post zavedl problém PCP pomocí dvou uspořádaných množin slov. Jeho definice je velmi snadno pochopitelná, ale pro formulování a dokazování matematických tvrzení nepraktická. Budeme tedy nadále užívat výhradně následující algebraickou definici PCP. Jejich ekvivalenci nyní ukážeme.
Podle Postovy definice tvoří instanci PCP dvojice uspořádaných množin slov $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$. Prvky u_i a v_i jsou slova nad abecedou B pro všechna $i \in \{1, \ldots, n\}$. O množinách U, V chceme rozhodnout, zda existuje posloupnost indexů $i_1, i_2, \ldots, i_k$, že platí rovnost slov $u_{i_1}u_{i_2}\ldots u_{i_k} = v_{i_1}v_{i_2}\ldots v_{i_k}$.
Tyto množiny si lze představovat také pomocí homomorfismů. Definujeme dva homomorfismy $g, h : A^* \to B^*$ nad abecedou $A = \{a_1, \ldots, a_n\}$ s n písmeny, pomocí vztahů $g(a_i) = u_i$ a $h(a_i) = v_i$ pro $i = 1, 2, \ldots, n$. Potom existuje řešení PCP na množinách U, V, právě když existuje slovo $w \in A^+$ splňující $g(w) = h(w)$.

Definice. Nechť $g, h : A^* \to B^*$ jsou homomorfismy pro dvě abecedy A, B, $A = \{a_1, \ldots, a_n\}$ je abeceda s n písmeny. Dvojici (g, h) nazveme *instance PCP*. Existuje-li neprázdné slovo w takové, že $g(w) = h(w)$, nazveme w *řešením instance* (g, h).

Definice. Modifikací PCP vznikne *zobecněné PCP (GPCP, z angl. Generalized PCP)*. Čtveřici $((p_1, p_2), g, h, (s_1, s_2))$, kde $g, h : A^* \to B^*$ jsou homomorfismy a $p_1, p_2, s_1, s_2 \in B^*$, nazveme *instancí GPCP*. Potom opět slovo $w \in A^*$ splňující rovnost $p_1g(w)s_1 = p_2h(w)s_2$, nazýváme *řešením této instance.*

V GPCP připouštíme jako řešení i prázdné slovo, neboť nemusí být vždy splněno $p_1 s_1 = p_2 s_2$.

Definice. Homomorfismus $g : A^* \to B^*$ nazveme *markovaný*, pokud je nenulující a obraz každého písmene z A začíná jiným písmenem, tj. platí-li pro všechna různá $a, b \in A : g(a) \wedge g(b) = \epsilon$.
Pokud je alespoň jeden z homomorfismů g a h dané instance PCP nebo GPCP periodický, řekneme o této instanci, že je *periodická*, jinak je *neperiodická.*

Od této chvíle se budeme zabývat výhradně binárním PCP (resp. GPCP), tedy s dvouprvkovou výchozí abecedou $A = \{a, b\}$, přestože některá uvedená tvrzení platí i ve větší obecnosti. Slova z množiny B^* lze samozřejmě zakódovat výhradně písmeny a a b, bez újmy na obecnosti tedy budeme předpokládat, že $g, h : \{a, b\}^* \to \{a, b\}^*$.
Řešení instance I můžeme konstruovat buď od začátku nebo od konce. Sestrojování odzadu ale představuje přímou konstrukci řešení pro „zrcadlovou instanci“, neboli *reversní instanci* k instanci I, jak dokládá Lemma 1.6.

Definice. Pro instanci GPCP $I = ((p_1, p_2), g, h, (s_1, s_2))$ definujeme *reversní instanci* jako instanci $\overline{I} = ((\overline{s_1}, \overline{s_2}), \overline{g}, \overline{h}, (\overline{p_1}, \overline{p_2}))$, kde $\overline{g}$, resp. $\overline{h}$, je zrcadlový homomorfismus homomorfismu g, resp.h.

Lemma 1.6. *Slovo w je řešením instance I, právě když $\overline{w}$ je řešením $\overline{I}$.*

Důkaz. Nechť w je řešení I. Podle definice tedy

$$p_1 g(w) s_1 = p_2 h(w) s_2$$

$$\overline{p_1 g(w) s_1} = \overline{p_2 h(w) s_2}.$$

Z definice zrcadlových obrazů a rovností $\overline{g(w)} = \overline{g}(\overline{w})$, $\overline{h(w)} = \overline{h}(\overline{w})$ je tedy tato rovnost ekvivalentní s rovností

$$\overline{s_1}\ \overline{g(w)}\overline{p_1} = \overline{s_2}\overline{h(w)}\overline{p_2}.$$

$$\overline{s_1}\ \overline{g}(\overline{w})\overline{p_1} = \overline{s_2}\overline{h}(\overline{w})\overline{p_2}.$$

To znamená, že $\overline{w}$ je řešením $\overline{I}$. Opačnou implikaci dokážeme zpětným postupem. □

Poznatky z této části byly čerpány z článků o Postově korespondenčním problému, především z 5.

2. Posloupnost následníků - jádro algoritmu

Oproti postupu, který byl naznačen v úvodní kapitole, vysvětlíme nejprve podrobně funkci bloků a následníků, a teprve po nich rozebereme myšlenku předřešení sestrojovaných pomocí převisů. Mnoho pojmů, které se primárně vztahují k následníkům totiž poté v kapitole o kandidátech na řešení využijeme.
Hlavní myšlenkou při důkazu polynomiality algoritmu pro řešení PCP je převedení neperiodické instance na jednodušší instance, tzv. následníky. Tímto postupem vytvoříme orientovaný strom následníků. Pro každou instanci může existovat více následníků, pro které platí: instance má řešení (dostatečně dlouhé), právě když některý z následníků má řešení.
Procházení každé větve tohoto stromu následníků by polynomialitu algoritmu porušilo. Následníky ovšem můžeme zredukovat do jediného, pokud existuje ještě následník těchto následníků. Tento strom tedy po této redukci neobsahuje téměř žádná větvení (mohou se vyskytnout jen v poslední úrovni). V této kapitole budeme uvažovat pouze neperiodické a nenulující instance. Poznatky v této kapitole byly čerpány z článků [4] a [5]. V těchto článcích byla dokázána právě polynomialita algoritmu, který hledá řešení binárního PCP.

2.1 Markované homomorfismy

Rozhodování PCP v případě markovaných homomorfismů je snazší a rozhodnutí nemarkovaného páru na tento případ poté převedeme. Za to ovšem budeme muset zaplatit přechodem ke GPCP, jak uvidíme v další části.

Definice. Nechť (g, h) je instance PCP a g i h jsou markované homomorfismy. *Blokem instance* (g, h) nazveme dvojici slov (e, f) splňující:

1. $g(e) = h(f)$,
2. pokud existují neprázdná $e' \leq e$ a $f' \leq f$ splňující také $g(e') = h(f')$, pak $e = e'$ a $f = f'$ (neboli dvojice (e, f) je prefixově minimální pár mající vlastnost 1.).

Lemma 2.1. *Pro markované homomorfismy g a h a libovolné neprázdné slovo $p \in \{a, b\}^*$ existuje nejvýš jedna prefixově minimální dvojice slov (u, v), pro kterou platí $pg(u) = h(v)$. Obdobně pro každé neprázdné slovo q existuje nejvýš jedna prefixově minimální dvojice (u, v), že $g(u) = qh(v)$.*

Důkaz. První písmeno slova p nám jednoznačně určuje písmeno $\mathrm{pref}_1(v)$, neboť h je markovaný homomorfismus. Takto lze opět podle prvního písmene převisu určit pokračování, dokud nedostaneme slovo v', že $h(v') \geq p$.
Je-li $h(v') = p$, jsme hotovi, jinak převis určuje první písmeno slova u, neboť g je také markovaný. Vidíme, že existuje vždy nejvýš jedna možnost pokračování, aby byla zachována porovnatelnost slov $pg(u)$ a $h(v)$. Druhý případ je symetrický. □

Důsledek 2.2. *Existují nejvýš dva různé bloky instance (g, h). Navíc pokud (e, f) i (e', f') jsou bloky (g, h), pak $\mathrm{pref}_1(e) \neq \mathrm{pref}_1(e')$ a zároveň $\mathrm{pref}_1(f) \neq \mathrm{pref}_1(f')$.*

Důkaz. Bloky získáme jako v Lemmatu 2.1 volbami $p = g(a)$, resp. $p = g(b)$, a pak při převzetí označení z tohoto Lematu platí $(au, v) = (e, f)$, resp. $(bu, v) = (e', f')$. Pokud oba bloky existují, potom $\text{pref}_1(e) \neq \text{pref}_1(e')$ je zřejmé z jejich konstrukce a $\text{pref}_1(f) \neq \text{pref}_1(f')$ poté plyne z markovanosti h. □

Definice. Jsou-li (e, f), (e', f') dva různé bloky instance (g, h), definujeme *následníka instance* (g, h) jako (g_1, h_1) rovnostmi

$$\begin{aligned} g_1(a) &= e, \quad g_1(b) = e', \\ h_1(a) &= f, \quad h_1(b) = f'. \end{aligned}$$

Následující Lemma nám ukáže přímou souvislost mezi řešením dané instance a řešením jejího následníka. Při řešení PCP tak můžeme mezi těmito dvěma instancemi přecházet, aniž bychom ztratili o řešení nějakou informaci. Přitom později dokážeme, že řešení následníka je jednodušší než řešení původní instance.

Lemma 2.3. *Mějme homomorfismy (g, h) a následníka této dvojice (g_1, h_1).*
1. Je-li w_1 řešením instance (g_1, h_1), je $g_1(w_1) = h_1(w_1)$ řešením instance (g, h).
2. Je-li w řešením (g, h), lze (w, w) rozložit na posloupnost bloků (g, h) a instance (g_1, h_1) má řešení.

Důkaz. 1. Z definice následníka víme, že slovo $g_1(w_1)$ sestává ze slov e a e'. Stejně tak slovo $h_1(w_1)$ je složeno ze slov f a f', která se střídají analogicky podle $g_1(w_1)$. Ale $g(e) = h(f)$ a $g(e') = h(f')$, takže i $g(g_1(w_1)) = h(h_1(w_1)))$.
2. První písmeno slova w nám určí jednoznačně jeden z bloků, označme jej (e, f). Protože tento blok je dán minimálními slovy, jejichž obrazy se rovnají, musí nutně řešení w začínat slovem e a slovem f. Pokud $e = f$, jsme hotovi, pokud ne, převis slov e a f určuje následující blok, který musí navazovat. Slovo w je tedy posloupností bloků a následník má řešení - každý blok je obrazem jednoho písmene homomorfismů g_1, h_1. □

Z Lemmatu 2.3 tedy vidíme, že řešení následníka udává pořadí bloků v řešení původní instance. Zavedení tohoto pojmu je totiž motivováno tím, že lze řešení na bloky rozložit. Máme-li totiž řešení w, jistě musí platit, že

$$w = u_1 \dots u_k = v_1 \dots v_k$$

pro nějaké dvojice (u_i, v_i) splňující $g(u_i) = h(v_i)$. Tyto dvojice bereme prefixově minimální. Vidíme, že dvojice (u_i, v_i) odpovídají přesně definici bloků této instance.

Definice. *Sufixová složitost homomorfismu g*, $\sigma(g)$, je počet různých neprázdných slov, která se objevují jako nějaký sufix obrazu g, formálně tedy

$$\sigma(g) = |\{x \neq \epsilon | \exists v \in \{a, b\}^* : g(a) = vx \text{ nebo } g(b) = vx\}|.$$

Víme, že existence řešení, stejně jako jeho struktura, pro dvojici (g, h) přímo souvisí s existencí a strukturou řešení pro jeho následníka (g_1, h_1). Právě sufixová složitost nám pomůže nahlédnout, že tento řetězec následníků máme pod kontrolou, dokážeme, že sufixová složitost následníků neroste. Z její definice vidíme, že sufixová složitost vlastně omezuje počet písmen v každém slově bloku dané instance.

Definice. Symbolem $\mathcal{O}_g$ budeme značit množinu *přesahů*, tj. neprázdných slov $u \in \text{suff}(g(a)) \cup \text{suff}(g(b))$, ke kterým existují (možná prázdná) slova e_1, e_2, f_1 a f_2, že $g(e_1) = h(f_1)u$ a $ug(e_2) = h(f_2)$. Obdobným způsobem zavedeme množinu $\mathcal{O}_h$.

Jak jsme dokázali v Lemmatu 2.1, každé $u \in \mathcal{O}_g$ jednoznačně určuje slovo f_2, které je sufixem slova f nebo f', kde (e, f), (e', f') jsou bloky, a splňuje $ug(e_2) = h(f_2)$ pro nějaké e_2. Toto zobrazení označíme $\pi_g : \mathcal{O}_g \to \text{suff}(f) \cup \text{suff}(f')$.

Lemma 2.4. *Zobrazení π_g je surjektivní.*

Důkaz. Vezměme libovolné slovo $v \in \text{suff}(f) \cup \text{suff}(f')$, nechť je sufixem slova f. Protože (e, f) je blok, víme, že platí $g(e) = h(f) = h(fv^{-1}v) = h(fv^{-1})h(v)$. Hledáme tedy takové slovo u, že $ug(e_2) = h(v)$ pro nějaké e_2. Rozložíme slovo e na $e = e_1e_2$ tak, aby $|h(fv^{-1})| < |g(e_1)|$ a e_1 bylo nejkratší takové. Poslední rovnost vynásobíme zleva inverzním slovem $h(fv^{-1})^{-1}$, takže dostaneme

$$h(fv^{-1})^{-1}g(e_1)g(e_2) = h(v).$$

Nyní je vidět, že volbou $u = h(fv^{-1})^{-1}g(e_1)$ požadavek splníme: u bude neprázdné, neboť nerovnost při volbě e_1 je ostrá, navíc z důvodu minimality bude sufixem $g(a)$ nebo $g(b)$. Také pro $f_1 = fv^{-1}$ dostaneme

$$g(e_1) = h(f_1)h(f_1)^{-1}g(e_1) = h(f_1)u,$$

což je poslední podmínka pro $u \in \mathcal{O}_g$. □

Důsledek 2.5. *Platí $\sigma(h_1) \leq |\mathcal{O}_g| \leq \sigma(g)$ a $\sigma(g_1) \leq |\mathcal{O}_h| \leq \sigma(h)$. Navíc rovnost $\sigma(h_1) = |\mathcal{O}_g|$ dostaneme, právě když je zobrazení π_g injektivní. Druhá rovnost $|\mathcal{O}_g| = \sigma(g)$ platí, právě když všechny sufixy $g(a)$ i $g(b)$ patří do $\mathcal{O}_g$. Obdobně i pro rovnosti s $\mathcal{O}_h$.*

Sufixová složitost následníka tedy není vyšší než u původní instance. Pokud by u každé generace sufixová složitost klesla (a bude-li existovat dostatečný počet generací následníků), bude sufixová složitost poslední generace rovna 2. To znamená, že homomorfismus g má jediný přesah, tedy $|g(a)| = |g(b)| = 1$ a totéž platí i o homomorfismu h. Rozhodnutí PCP u takových instancí již není těžké.

Definice. Řekneme, že dvojice homomorfismů (g, h) má *stabilní sufixovou složitost*, jestliže má takového následníka (g_1, h_1), že $\sigma(g) = \sigma(h_1)$ a $\sigma(h) = \sigma(g_1)$.

Právě instance se stabilní sufixovou složitostí mohou při hledání polynomiálního algoritmu způsobovat problémy. Jejich následník totiž není jednodušší z pohledu sufixové složitosti a převedení problému z dané instance na jejího následníka by nám při hledání řešení příliš nepomohlo. Musíme tedy tyto instance umět přesně poznat.
Pro další potřeby označíme g_a a g_b vždy jedno ze slov $g(a)$, $g(b)$ tak, aby bylo $\text{pref}_1(g_a) = a$ a $\text{pref}_1(g_b) = b$. Stejně tak i pro druhý homomorfismus h. Díky symetrii neztratíme z obecnosti, budeme-li předpokládat, že $g_a = g(a)$ a $g_b = g(b)$. Pro h ale už takový předpoklad udělat nelze. Také můžeme ztotožnit dvojice homomorfismů, které se liší pouze záměnou a a b anebo g a h.

Definice. Nechť $\mu : \{a,b\}^* \to \{a,b\}^*$ značí homomorfismus, který přehazuje písmena a a b, tedy $\mu(a) = b$, $\mu(b) = a$, identitu na množině $\{a,b\}^*$ označíme id. Pak dvojice homomorfismů (g,h) a (g',h') nazveme *symetrické*, pokud je

$$\{g,h\} = \{\mu_1 \circ g' \circ \mu_2, \mu_1 \circ h' \circ \mu_2\},$$

kde $\mu_1, \mu_2 \in \{\mu, id\}$.

Následující tvrzení budeme dokazovat vždy jen pro jeden ze všech symetrických párů, důkaz pro ostatní páry s ním symetrické bývá již jednoduchý. Tato tvrzení jsou ovšem klíčová k důkazu polynomiální složitosti celého algoritmu.

Lemma 2.6. *Nechť (g,h) má stabilní sufixovou složitost, u je sufix g_a nebo g_b. Je-li c první písmeno slova u, pak u je prefixově porovnatelné s h_c. Navíc existuje slovo v prefix g_a nebo g_b, pro který $vu \in \{g_a, g_b\}$ a v je sufixově porovnatelné s h_a nebo h_b.*

Důkaz. Pár (g,h) má stabilní sufixovou složitost, proto z Důsledku 2.5 máme $u \in \mathcal{O}_g$, což dokazuje první část lemmatu. Druhá část je zřejmá. □

Lemma 2.7. *Nechť dvojice homomorfismů (g,h) má stabilní sufixovou složitost a poslední písmeno h_a i h_b je a. Pak $g_a = a^k$ pro nějaké $k \geq 1$ a $g_b = ba^l$, $l \geq 0$.*

Důkaz. Označme n největší číslo, že ba^n je sufix g_a nebo g_b. Protože potom platí $a^n \in \mathcal{O}_g$, z Lemmatu 2.6 nalezneme (i prázdné) slovo v, pro které $va^n \in \{g_a, g_b\}$ a v je sufixově porovnatelné s h_a nebo h_b. Kdyby v bylo neprázdné, musí být tedy písmeno a jeho posledním písmenem, což je ale spor s maximalitou n. Proto $v = \epsilon$, a tedy $g_a = a^n$.
Předpokládejme pro spor, že $g_b = bu$, $u \neq a^l$ pro každé $0 \leq l \leq n$. Pak $u \in \mathcal{O}_g$ a přitom u není sufix g_a. Proto musí existovat slova e_1, f_1, že platí

$$g(e_1 b) = h(f_1)u,$$

$$g(e_1)bu = h(f_1)u.$$

Z toho vyplývá, že poslední písmeno slova $h(f_1)$ je b, což je ve sporu s předpokladem. □

Tvrzení 2.8. *Nechť (g,h) je dvojice binárních markovaných homomorfismů. Sufixová složitost (g,h) je stabilní, právě když nastane jeden z následujících případů (až na symetrii):*

	$g(a)$ $h(x)$	$g(b)$ $h(y)$	$g_1(a)$ $h_1(a)$	$g_1(b)$ $h_1(b)$
(1)	a au	b bv	au x	bv y
	u, v libovolná slova			
(2)	a^j a^l	b^k b^m	a^l x^j	b^m y^k
	$(j,l) = (k,m) = 1$			

	$g(a)$ $h(x)$	$g(b)$ $h(y)$	$g_1(a)$ $h_1(a)$	$g_1(b)$ $h_1(b)$
(3)	ab^j ab^l	b^k b^m	$ab^{j'}$ $xy^{l'}$	b^m y^k
	$(k,m)=1$, $j \le k$		$l \le m$, $j + j'k = l + l'm$	
(4)	$(ab)^j$ $(ab)^l a$	$(ba)^k$ $(ba)^m b$	a^{l+m+1} $(xy)^j$	b^{l+m+1} $(yx)^k$
	$(j, l+m+1)=1$ $(k, l+m+1)=1$			
(5a)	$(ab)^j a$ $(ab)^l a$	$(ba)^k b$ $(ba)^m b$	$(ab)^{j'} a$ $(xy)^{l'} x$	$(ba)^{k'} b$ $(yx)^{m'} y$
	$(j+k+1, l+m+1)=1$		$j'(j+k+1) + j = l'(l+m+1) + l$, $k'(k+j+1) + k = m'(l+m+1) + m$	
(5b)	$(ab)^j a$ $(ab)^l a$	$(ba)^k b$ $(ba)^m b$	$(ab)^{(l+m+1)/2}$ $(xy)^{(j+k+1)/2}$	$(ba)^{(l+m+1)/2}$ $(yx)^{(j+k+1)/2}$
	$(j+k+1, l+m+1)=2$ $j-l$ liché			
(6)	a a	$(ba^l)^k b$ $(ba^l)^m b$	a x	$(ba^l)^m b$ $(yx^l)^k y$
	$(k+1, m+1)=1$			
(7)	a $a^j b^l$	b^k b^m	$a^j b^{j'}$ $xy^{l'}$	b^m y^k
	$l \le m$ a $(k,m)=1$		$j'k = l + l'm$	

Tabulka 2.1: **Případy se stabilní sufixovou složitostí.**
Indexy j', l', k' a m' jsou nejmenší nezáporná čísla splňující uvedený vztah.

Důkaz. Důkaz provedeme rozborem případů, vynecháme ovšem případy, které jsou symetrické s uvedenými. Důležitým poznatkem je, že ke stálé sufixové složitosti je potřeba, aby každý sufix byl zároveň přesahem, tj. například je-li aab sufixem $g(a)$ nebo $g(b)$, musí slovo h_a začínat písmeny aa (pokud není délky 1).
1. Pokud jeden z homomorfismů zachovává délku. Bez újmy na obecnosti nechť $g = id$, $h_a = au$, $h_b = bv$ pro nějaká (možná prázdná) slova u, v. Pak následníkem je dvojice (g_1, h_1), přitom $h_1 = g$ a $g_1 = h$, pokud $h(a) = h_a$, v druhém případě jsou navíc zaměněna písmena a a b. Sufixová složitost je tedy stabilní.
Nechť je dále vždy alespoň jeden z obrazů každého z homomorfismů g a h délky alespoň 2. Označíme x a y taková různá písmena, že $h(x) = h_a$, $h(y) = h_b$. Také pro přehlednost budeme značit největší společný dělitel hodnot m a n jako (m, n). Předpokládejme, že sufixová složitost (g, h) je stabilní, dokážeme, že musí být jednoho z uvedených tvarů.
2. Nechť všechna slova g_a, g_b, h_a, h_b mají délku alespoň 2.
2.1. Nechť $aa \le g_a$ a $bb \le g_b$. Z Lemmatu 2.6 plyne, že $g_a, h_a \in a^+$ a $g_b, h_b \in b^+$. Označíme

$$\begin{aligned} g(a) &= a^j, \quad g(b) = b^k, \\ h(x) &= a^l, \quad h(y) = b^m. \end{aligned}$$

To znamená, že sufixová složitost homomorfismu g je $\sigma(g) = j+k$, je to totiž počet všech sufixů $g(a)$ nebo $g(b)$, které jsou různé, sufixová složitost homomorfismu h

je $\sigma(h) = l+m$. Není těžké nahlédnout, že následník (g_1, h_1) tohoto páru existuje a je tvaru

$$\begin{aligned} g_1(a) &= a^{l/(j,l)}, \quad g_1(b) = b^{m/(k,m)}, \\ h_1(a) &= x^{j/(j,l)}, \quad h(b) = y^{k/(k,m)}, \end{aligned}$$

takže sufixová složitost je stabilní, právě když $(j,l) = (k,m) = 1$ (případ (2)).
2.2. Nechť $ab \leq g_a$, $bb \leq g_b$. Z Lemmatu 2.6 dostaneme, že slova $g_b, h_b \in b^+$ a $g_a, h_a \in ab^+$. Máme tedy

$$\begin{aligned} g(a) &= ab^j, \quad g(b) = b^k, \\ h(x) &= ab^l, \quad h(y) = b^m. \end{aligned}$$

Následník tohoto prvku je tvaru

$$\begin{aligned} g_1(a) &= ab^{j'}, \quad g_1(b) = b^m, \\ h_1(a) &= xy^{l'}, \quad h_1(b) = y^k, \end{aligned}$$

kde j' a l' jsou nejmenší nezáporná čísla splňující $j + j'k = l + l'm$. Slovo b je přesahem g, aby byla sufixová složitost stabilní, musí existovat slova e_1, f_1 taková, že platí $bg(e_1) = h(f_1)$. Pomocí jednoduché teorie čísel lze dokázat, že taková slova existují, právě když je $(k,m) = 1$. Pokud $j > k$, je $\sigma(g) = j+1$. Protože $\sigma(h_1) = \max(l', k) + 1$ a sufixová složitost má být stabilní, je $l' = j$. Z rovnice $j + j'k = l + l'm$ vidíme, že $j' \geq m$. Potom ale uvedenou rovnici splňují i čísla $j' - m$ a $j - k$ místo j' a l', která jsou menší a nezáporná, což je spor s minimalitou j', l'. Obdobně dostaneme spor i pro $l > m$.
Proto $j \leq k$ a $l \leq m$, tyto podmínky odpovídají případu (3). Sufixová složitost je potom stabilní, neboť $\sigma(g) = \sigma(h_1) = k+1$ a $\sigma(h) = \sigma(g_1) = m+1$.
2.3. Nechť $ab \leq g_a$ a $ba \leq g_b$. Pak znovu z Lemmatu 2.6 máme $g_a, h_a \in (ab)^+$ a $g_b, h_b \in (ba)^+$. Z Lemmatu 2.7 máme $\mathrm{suff}_1(h_a) \neq \mathrm{suff}_1(h_b)$ a $\mathrm{suff}_1(g_a) \neq \mathrm{suff}_1(g_b)$.
2.3.1. Jsou-li g, h dané

$$\begin{aligned} g(a) &= (ab)^j, \quad g(b) = (ba)^k, \\ h(x) &= (ab)^l, \quad h(y) = (ba)^m, \end{aligned}$$

nemohou být všechny sufixy liché délky zároveň přesahy, takže tento pár nemá stabilní sufixovou složitost.
2.3.2. Mějme j, k, l, m taková, že

$$\begin{aligned} g(a) &= (ab)^j, \quad g(b) = (ba)^k, \\ h(x) &= (ab)^l a, \quad h(y) = (ba)^m b, \end{aligned}$$

přitom $j, k > 0$ a $l, m \geq 0$. Potom existuje následník této instance a má tvar

$$\begin{aligned} g_1(a) &= a^{(l+m+1)/(j,l+m+1)}, \quad g_1(b) = b^{(l+m+1)/(k,l+m+1)}, \\ h_1(a) &= (xy)^{j/(j,l+m+1)}, \quad h_1(b) = (yx)^{k/(k,l+m+1)}. \end{aligned}$$

Takže sufixové složitosti původních homomorfismů jsou $\sigma(g) = 2(j+k)$ a také $\sigma(h) = 2(l+m+1)$, zatímco následnických

$$\sigma(g_1) = \frac{l+m+1}{(j, l+m+1)} + \frac{l+m+1}{(k, l+m+1)}, \ \sigma(h_1) = \frac{2j}{(j, l+m+1)} + \frac{2k}{(k, l+m+1)}.$$

To znamená, že sufixová složitost páru (g,h) je stabilní, právě když je splněna rovnost $(j, l+m+1) = (k, l+m+1)$. Dostáváme se tedy k případu (4).

2.3.3. Poslední možností (až na symetrické případy) je pár (g, h) zadaný

$$\begin{aligned} g(a) &= (ab)^j a, \quad g(b) = (ba)^k b, \\ h(x) &= (ab)^l a, \quad h(y) = (ba)^m b, \end{aligned}$$

pro nějaká j, k, l, m. Položme $r = \frac{l+m+1}{(j+k+1, l+m+1)}$ a $s = \frac{j+k+1}{(j+k+1, l+m+1)}$. To jsou jistě nejmenší kladná čísla, která splňují $g((ab)^r) = h((xy)^s)$ a také nejmenší taková, že platí $g((ba)^r) = h((yx)^s)$. Dvojice slov $((ab)^r, (xy)^s)$ a $((ba)^r, (yx)^s)$ jsou buď bloky, nebo se na bloky rozloží. V prvním případě tedy

$$\begin{aligned} g_1(a) &= (ab)^r, \quad g_1(b) = (ba)^r, \\ h_1(a) &= (xy)^s, \quad h_1(b) = (yx)^s, \end{aligned}$$

ve druhém bude

$$\begin{aligned} g_1(a) &= (ab)^{j'} a, \quad g_1(b) = (ba)^{k'} b, \\ h_1(a) &= (xy)^{l'} x, \quad h_1(b) = (yx)^{m'} y, \end{aligned}$$

kde j', k', l', m' jsou nejmenší nezáporná čísla, splňující

$$j'(j + k + 1) + j = l'(l + m + 1) + l \tag{2.1}$$

$$k'(j + k + 1) + k = m'(l + m + 1) + m \tag{2.2}$$

Sufixová složitost g je $2(j + k + 1)$, sufixová složitost h_1 je v prvním případě $4s$, ve druhém $2s$. Protože chceme, aby sufixová složitost byla stabilní, musí být $(j + k + 1, l + m + 1) \leq 2$, neboť $s = \frac{j+k+1}{(j+k+1, l+m+1)}$.
Jsou-li tedy hodnoty $j+k+1$ a $l+m+1$ nesoudělné, má rovnice (2.1) nezáporné řešení $j' < l + m + 1$, $l' < j + k + 1$ a dostáváme první typ následníka. Sufixová složitost je v tomto případě stabilní a jde o případ (5a).
Nyní předpokládejme, že $(j + k + 1, l + m + 1) = 2$. Stabilní sufixovou složitost dostaneme, právě když rovnice (2.1) nemá řešení, což nastává, právě když $j - l$ je liché, tedy v případě (5b).
3. Nechť $g_a = a$ a zbylá tři slova mají délku alespoň 2.
3.1. Předpokládejme, že $bb \leq g_b$. Pak z Lemmatu 2.6 máme

$$\begin{aligned} g(a) &= a, \quad g(b) = b^k, \\ h(x) &= a^j b^l, \quad h(y) = b^m, \end{aligned}$$

pro nějaké $j \geq 1$, $k, m \geq 2$ a $j + l \geq 2$. Pokud $l = 0$, dostaneme se opět k případu (2). Pro $l > 0$ můžeme argumentovat stejně jako v 2.2., sufixová složitost je stabilní, právě když $(k, m) = 1$ a $l \leq m$, tedy případ (7).
3.2. Nechť $ba \leq g_b$, potom $ba \leq h_b$.
3.2.1. Je-li $aa \leq h_a$, pak podle Lemmatu 2.6 je $g_b \in ba^+$ a z Lemmatu 2.7 dostaneme $h_a \in a^+$. Tato situace je symetrická s případem (3).
3.2.2. Z předpokladu $ab \leq h_a$ a Lemmatu 2.6 plyne, že $g_b \leq (ba)^+$. Lemma 2.7 nám dává $g_b \in (ba)^+ b$. Celkově tedy $bab \leq g_b$, takže ani h_a ani h_b neobsahují slova baa ani bb jako podslova. Proto $g_a, h_a \leq (ab)^+$ a $g_b, h_b \leq (ba)^+$. O těchto případech jsme již ukázali, že vedou k případům (4) resp. (5).
4. Mějme $g_a = h_a = a$, g_b, h_b délky alespoň dvě.
4.1. Pro $bb \leq g_b$ nebo $bb \leq h_b$ se dostaneme opět k případu (2).

4.2. Je-li poslední písmeno slova g_b nebo h_b písmeno a, pak dostaneme případ symetrický s případem (3).
4.3. Nechť $ba \leq g_b$ a $ba \leq h_b$ a poslední písmeno obou těchto slov je b. Potom díky Lemmatu 2.6 víme, že bb není faktor g_b ani h_b a máme

$$g_b = ba^{k_1}ba^{k_2}\cdots ba^{k_m}b,$$

$$h_b = ba^{l_1}ba^{l_2}\cdots ba^{l_{m'}}b.$$

Navíc z tohoto Lemmatu také vyplývá, že $k_i = l_1$ pro všechna $i = 1, \cdots, m$ a $l_j = k_1$ pro každé $j = 1, \cdots, m'$. Proto

$$\begin{aligned} g(a) &= a, \quad g(b) = ((ba)^t)^k b, \\ h(x) &= a, \quad h(y) = ((ba)^t)^m b. \end{aligned}$$

Obdobně jako v případě 2.3.3. je sufixová složitost stabilní, právě když platí $(k+1, m+1) = 1$, případ (6).
5. Poslední případ, který zbývá, je $g_a = a$ a $h_b = b$.
5.1. Je-li $bb \leq g_b$, je $h_a = a^j b^l$ podle Lemmatu 2.6.
5.1.1. Pokud navíc $l = 0$, bude $aa \leq h_a$ a $g_b = b^t a^m$. Dostaneme situaci symetrickou s případem (7), tuto jsme již prozkoumali v části 3.1.
5.1.2. Je-li $l > 0$, pak z Lemmatu 2.7 plyne, že $g_b \in b^+$, obdržíme tedy opět případ (7).
5.2. Máme-li $ba \leq g_b$, pak tedy $ab \leq h_a$. Přitom pokud by jedno z těchto slov bylo délky právě dva, dostaneme spor s Lemmatem 2.7. Proto $bab \leq g_b$ a $aba \leq h_a$, dostáváme se tak k případu (5). □

Zamysleme se nyní nad tím, jak bude probíhat algoritmus pro řešení PCP. Nejprve můžeme řešení konstruovat podle převisů. Pokud však zjistíme, že daná instance má oba bloky, tj. má následníka, mohlo by dojít k mnoha situacím, kdy je potřeba vyzkoušet obě písmena. To nám ovšem navyšuje časovou složitost algoritmu. Proto se snažíme v těchto případech nahradit vstupní instanci jejím následníkem, jehož sufixová složitost podle Důsledku 2.5 neroste.
Pokud ovšem obdržíme jednu z instancí z Tvrzení 2.8, sufixová složitost následníka ani neklesne. Omezíme-li se pouze na tzv. balancované páry homomorfismů, dokážeme, že sufixová složitost zůstává stabilní nejvýš pro jednu generaci následníků v řadě (až na pár výjimek) - Tvrzení 2.9. U nebalancovaných homomorfismů později nalezneme omezení na délku řešení a u výjimečných případů, tedy těch se stálou sufixovou složitostí, ukážeme, jak jednoduše rozhodnout o jejich řešení.

Definice. Řekneme, že instance (g, h) *má stálou sufixovou složitost*, jestliže existuje následník (g_1, h_1) této instance a také existuje jeho následník (g_2, h_2), že $\sigma(g) = \sigma(h_1) = \sigma(g_2)$ a $\sigma(h) = \sigma(g_1) = \sigma(h_2)$.

Definice. Označíme $|w|_a$, resp. $|w|_b$, počet písmen a, resp. b, ve slově w. Řekneme, že pár (g, h) je *balancovaný*, pokud pro všechna $\rho(.) \in \{|.|, |.|_a, |.|_b\}$ platí jedna z možností:

$$\begin{array}{lllll} (1) & \rho(g(a)) < \rho(h(a)) & \text{a} & \rho(g(b)) > \rho(h(b)) & \text{nebo} \\ (2) & \rho(g(a)) > \rho(h(a)) & \text{a} & \rho(g(b)) < \rho(h(b)) & \text{nebo} \\ (3) & \rho(g(a)) = \rho(h(a)) & \text{a} & \rho(g(b)) = \rho(h(b)). & \end{array}$$

V opačném případě nazýváme pár *nebalancovaný*.

Tvrzení 2.9. *Nechť máme balancované páry (g_i, h_i) pro $i = 0, 1, 2$ takové, že (g_1, h_1) je následníkem (g_0, h_0) a (g_2, h_2) je následníkem (g_1, h_1). Předpokládejme, že všechny tři mají stejnou sufixovou složitost. Pak buď g_1 i h_1 zachovávají délku, nebo platí*

$$g_0(a) = a^j,\ g_0(b) = b^k,$$
$$h_0(a) = b^m,\ h_0(b) = a^l,$$

pro celá $j, k, l, m > 0$ splňující podmínku

$$NSD(j, m) = NSD(k, l) = NSD(m, k) = NSD(l, j) = 1.$$

Bez újmy na obecnosti předpokládejme, že $m = min\ \{j, k, l, m\}$. Pak bude platit $m < k < l$ a $m < j < l$. Posloupnost následníků $((g_i, h_i))_{i\in\mathbb{N}_0}$ je dobře definovaná.

Důkaz. Páry (g_0, h_0) se stabilní sufixovou složitostí popisuje Tvrzení 2.8, zbývá dokázat, že pouze případy (1) a (2) mohou vést ke stejné sufixové složitosti i u dalších dvou generací balancovaných následníků.
Předpokládejme nejdřív, že g_1 i h_1 zachovávají délku, potom $\sigma(g_1) = \sigma(h_1) = 2$ a podle předpokladů tedy také $\sigma(g_0) = \sigma(h_0) = 2$ (to ovšem nemusí nutně znamenat, že (g_0, h_0) také zachovává délku). Následník páru (g_1, h_1) je ale stejný, tj. $(g_2, h_2) = (g_1, h_1)$, takže má stejnou sufixovou složitost a posloupnost následníků je nekonečná a korektně definovaná.
Dále tedy budeme předpokládat, že g_1 nezachovává délku. Protože je pár (g_1, h_1) balancovaný (a nenulující), nemůže ani h_1 zachovávat délku. Rozebereme všechny případy se sufixovou složitostí popsané v Tvrzení 2.8. Označíme pro jednoduchost $(g, h) = (g_0, h_0)$ a homomorfismus μ daný $\mu(a) = b$, $\mu(b) = a$. Také předpokládejme, že $g(a) = g_a$.
Případ (1): Nechť $g = id$. Dvojice (g_1, h_1) je tvaru $(h \circ \mu, \mu)$, pokud $h_a = h(a)$, v opačném případě je tvaru (h, id). Obě možnosti jsou ale ve sporu s předpokladem, že g_1 ani h_1 nezachovávají délku.
Případ (2): Protože (g, h) je balancovaný, musí být $h_0(a) = b^m$ a $h_0(b) = a^l$. Potom je $g_1(a) = a^l$, $g_1(b) = b^m$ a $h_1(a) = b^j$, $h_1(b) = a^k$, a protože má (g_1, h_1) stabilní sufixovou složitost, je i $\mathrm{NSD}(j, m) = \mathrm{NSD}(k, l) = 1$. Další následník má tvar $g_2(a) = a^k$, $g_2(b) = b^j$ a $h_2(a) = b^l$, $h_2(b) = a^m$ a následující pár je shodný s první dvojicí homomorfismů, tj. $(g_3, h_3) = (g_0, h_0)$. Posloupnost následníků má tedy periodu čtyři. Protože $m = \min\{j, k, l, m\}$ a (g, h) je balancovaný, musí být $m < j$ a $k < l$. Z balancovanosti páru (g_1, h_1) navíc plyne $m < k$ a $j < l$, takže platí i nerovnosti uvedené v závěru tvrzení.
Případ (3): Je-li $j = l$, potom i $k = m$, aby byl první následník balancovaný. Pak ale buď $j = l = 0$ a oba homomorfismy zachovávají délku, nebo $j = l = 1$. V druhém případě zachovávají délku g_1 i h_1, což jsme již také vyloučili.
Nechť tedy $j < l$. Uvažujme nejprve případ, kdy $h(a) = b^m$, $h(b) = ab^l$. Následník má potom tvar $g_1(a) = ab^{j'}$, $g_1(b) = b^m$ a $h_1(a) = ba^{l'}$, $h_1(b) = a^k$, jeho balancovanost tedy zajišťuje pouze volba $j' = l' = 0$. Pak nutně $m = k$, takže oba homomorfismy by zachovávali délku.
Máme-li druhý případ, tedy $h(a) = ab^l$, $h(b) = b^m$, je z balancovanosti a předpokladu $j < l$ splněno $m < k$. Následník má potom tvar $g_1(a) = ab^{j'}$, $g_1(b) = b^m$ a $h_1(a) = ab^{l'}$, $h_1(b) = b^k$, přitom platí $j' \leq l'$, takže z balancovanosti párů (g_1, h_1) je $k \leq m$, což je spor.

Případ (4): Kdyby $l = 0$, nutně z balancovanosti (g_0, h_0) a předpokladu, že $j, k > 0$, plyne, že $m \neq 0$. Obdobně z $m = 0$ plyne $l \neq 0$, takže $l + m > 0$. Následník má tvar $g_1(a) = a^{l+m+1}$, $g_1(b) = b^{l+m+1}$ a $h_1(a) = (ab)^j$, $h_1(b) = (ba)^k$ pokud $h_a = h(a)$, ve druhém případě $h_1(a) = (ba)^j$, $h_1(b) = (ab)^k$. Následník tohoto páru tedy nemůže existovat.
Případ (5a): Je-li $j = l$, nutně $k = m$ a oba homomorfismy musí zachovávat délku. Proto předpokládejme bez újmy na obecnosti, že $j < l$ a $k > m$, opačný případ je symetrický. Kdyby $j + k = l + m$, neměl by pár (g_0, h_0) stabilní sufixovou složitost. Proto předpokládejme $j + k > l + m$. Následník (g_1, h_1) má tvar $g_1(a) = (ab)^{j'}a$, $g_1(b) = (ba)^{k'}b$, $h_1(a) = (xy)^{l'}x$ a $h_1(b) = (yx)^{m'}y$, kde $x = a$, $y = b$, pokud $h(a) = h_a$, jinak $x = b$ a $y = a$. Přitom je splněno

$$j'(j + k + 1) + j = l'(l + m + 1) + l,$$

$$k'(j + k + 1) + k = m'(l + m + 1) + m.$$

Navíc můžeme bez újmy na obecnosti předpokládat, že je splněno $l' < j'$ a $k' < m'$. Pak tedy platí

$$l - j = j'(jk + 1) - l'(l + m + 1).$$

Pak ale levá strana je záporná a pravá kladná, což nelze. Pokud bychom uvažovali $j + k < l + m$, lze ke sporu dospět obdobně.
Případ (5b): Lze nahlédnout, že (g_1, h_1) nemá stabilní sufixovou složitost.
Případ (6): Dvojice (g_0, h_0) je balancovaná, pouze pokud $k = m$. Potom buď $k = m = 0$ a oba homomorfismy zachovávají délku, nebo $k = m > 0$ a sufixová složitost (g_0, h_0) není stabilní.
Případ (7): Pro $t > 1$ není dvojice (g_0, h_0) balancovaná, je-li $t = 1$, dostaneme se k případu (3).

□

2.2 Vztah markovaných a nemarkovaných homomorfismů

Na začátku této části připomeňme, že prozatím uvažujeme pouze instance s nenulujícími a neperiodickými homomorfismy. Pokusíme se nyní vysvětlit, jak vypadají bloky nemarkovaných instancí, jejich následnické instance a jak se od nemarkovaného homomorfismu dostat k jeho markované verzi.
Z Lemmatu 1.5 vyplývá, že pokud je $uv = vu$, potom u a v jsou mocniny stejného slova. Kdyby $g(ab) = g(ba)$, tj. $g(a)g(b) = g(b)g(a)$, pak tedy $g(a)$ a $g(b)$ jsou mocninami téhož slova, což odporuje předpokladu, že g je neperiodický homomorfismus, proto $g(ab) \neq g(ba)$.

Definice. Pro neperiodický homomorfismus g zavedme jeho *nutný začátek* jako slovo $z_g = g(ab) \wedge g(ba)$.

Z definice vidíme, že slovem z_g začíná obraz všech slov, jejichž první písmena jsou různá. Další Lemma 2.10 nám navíc ukáže, že je to začátek obrazu všech dostatečně dlouhých slov.

Lemma 2.10. *Nechť au a bv jsou dvě slova taková, že platí $|g(au)| > |z_g|$ a $|g(bv)| > |z_g|$. Pak $g(au) \wedge g(bv) = z_g$.*

Důkaz. Označme c a d ta písmena, že $z_g c$ je prefix $g(ab)$ a $z_g d$ prefix $g(ba)$. Z definice z_g vidíme, že $c \neq d$. Indukcí dokážeme, že $z_g c$ je prefixem $g(a^i b)$ pro každé $i \geq 1$.
Pro $i = 1$ je to splněno z volby písmena c.
Nechť $i > 1$ a tvrzení platí až do $i-1$. Z definice tedy platí nerovnosti $z_g c \leq g(aba)$ a $g(a)z_g \leq g(aba)$ a protože $|z_g c| \leq |g(a)z_g|$, je splněno

$$z_g c \leq g(a)z_g. \tag{2.3}$$

Úpravami indukčního předpokladu získáme

$$z_g c \leq g(a^{i-1}b)$$

$$g(a)z_g c \leq g(a)g(a^{i-1}b)$$

a přidáním (2.3) tedy máme $z_g c \leq g(a)z_g c \leq g(a^i b)$. Proto $z_g c \leq g(au)$.
Důkaz, že $z_g d$ je prefixem $g(bv)$, je obdobný a tvrzení je tedy dokázáno. □

Toto lemma jinými slovy říká, že pro každé dostatečně dlouhé slovo w platí $z_g \leq g(w)$. Dokonce pro jakékoliv slovo w platí $z_g \bowtie g(w)$. Slovo z_g je tedy jakýsi nutný začátek obrazu všech slov.

Důsledek 2.11. *První písmeno každého slova w je tedy dané jakýmkoliv prefixem slova $g(w)$, který je ostře delší než $|z_g|$, neboli je určené $(|z_g|+1)$-ním písmenem $g(w)$. Proto také pro jakákoliv slova u_1, u_2 máme*

$$g(u_1)z_g \wedge g(u_2)z_g = g(u_1 \wedge u_2)z_g,$$

a tudíž platí $u_1 \leq u_2$, právě když $g(u_1)z_g \leq g(u_2)z_g$.

Na příkladech ukážeme, že z_g může být kratší něž obrazy obou písmen, delší než obraz jednoho z písmen, a dokonce i delší než oba obrazy.

Příklad 3. Je-li $g_1(a) = aba$, $g_1(b) = aa$, potom

$$z_{g_1} = g_1(ab) \wedge g_1(ba) = abaaa \wedge aaaba = a,$$

je tedy vlastním prefixem $g_1(a)$ i $g_1(b)$.

Příklad 4. Pro $g_2(a) = abaa$, $g_2(b) = ab$ máme

$$z_{g_2} = abaaab \wedge ababaa = aba,$$

takže $g_2(a) > z_{g_2}$ a $z_{g_2} > g_2(b)$.

Příklad 5. Máme-li $g_3(a) = aaba$, $g_3(b) = aab$, bude

$$z_{g_3} = aabaaab \wedge aabaaba = aabaa,$$

a z_{g_3} je tedy ostře delší než obrazy obou písmen.

Na nutných prefixech z_g, z_h je také vidět výhoda binární abecedy. Už kdybychom měli třeba třípísmennou abecedu, můžeme mít problém s jejich definicí, jak ukazuje následující příklad.

Příklad 6. V případě abecedy $\{a, b, c\}$ a homomorfismu g daného rovnostmi

$$g(a) = aba, \quad g(b) = abb, \quad g(c) = acc$$

můžeme definovat z_g dvěma způsoby. Buď jako slovo a, které je společné začátku obrazů každého písmene, jenže pak budou slova $z_g^{-1}g(a)z_g$ a $z_g^{-1}g(b)z_g$ obě začínat písmenem b, což nechceme. Druhá možnost je definovat $z_g = ab$, tedy největší společný prefix dvou obrazů, ale tím zase nemusí začínat každé slovo $g(w)$. Pouze binární abeceda splňuje oba tyto naše požadavky.

Definice. Pokud jsou slova z_g a z_h sufixově porovnatelná, definujeme *kritický převis homomorfismů g a h* jako $c(g, h) = z_h z_g^{-1}$. Ze symetrie můžeme v plné obecnosti předpokládat, že z_g je sufix z_h, a tedy $c(g, h) \in A^*$.

Všimněme si, že v případě markovaných homomorfismů kritický přesah vždy existuje, je to totiž prázdné slovo. Důležitým faktem je, že pokud kritický převis neexistuje, tedy pokud z_g a z_h nejsou sufixově porovnatelná, nebude existovat řešení rozložitelné na bloky, jak jej definujeme v následující části. Název „kritický“ odpovídá situaci při prodlužování předřešení podle převisů, které bylo naznačeno v úvodní kapitole. Pokud totiž zkonstruujeme předřešení (w_1, w_2), které tvoří kritický převis, je splněno $g(w_1)z_g = h(w_2)z_h$ a nelze rozhodnout o prodloužení předřešení podle převisu jeho obrazů.

Přirozeným a jednoznačným způsobem nyní můžeme z každého homomorfismu vytvořit markovaný:

Definice. *Markovanou verzi* homomorfismu g definujeme jako homomorfismus g_m takový, že $g_m(a) = z_g^{-1}g(a)z_g$ a $g_m(b) = z_g^{-1}g(b)z_g$. Stejným způsobem definujeme i markovanou verzi homomorfismu h.

Z Lemmatu 2.10 plyne, že g_m je korektně definovaný homomorfismus a pro každé slovo $w \in A^*$ platí $g_m(w) = z_g^{-1}g(w)z_g$. V tomto tkví výhoda binární abecedy - stačí nám uvažovat pouze markované homomorfismy, neboť každý homomorfismus má jednoznačně přiřazenu markovanou verzi.

2.3 Následníci nemarkovaných homomorfismů

Naším cílem je pokusit se definovat bloky podobně jako u markovaných homomorfismů. Jak už jsme naznačili, budeme muset přejít ke zobecněnému PCP, proto všechna tvrzení formulujeme v obecnějším znění využitelném i u instancí GPCP.

Lemma 2.12. *Mějme dvojici homomorfismů (g, h), pro niž existuje kritický převis. Pro každé $c \in \{a, b\}$ existuje nejvýš jedna prefixově minimální dvojice neprázdných slov (e_c, f_c), že $pref_1(e_c) = c$ a zároveň*

$$c(g, h)g(e_c) = h(f_c)c(g, h).$$

Navíc existují-li obě dvojice (e_a, f_a) i (e_b, f_b), platí $pref_1(f_a) \neq pref_1(f_b)$.

Důkaz. Víme, že pro markované homomorfismy platí obdobné tvrzení, konkrétně Důsledek 2.2. Pro dané c tedy existuje nejvýš jeden prefixově minimální pár slov (e_c, f_c) takový, že

$$\begin{aligned} g_m(e_c) &= h_m(f_c) \\ z_g^{-1} g(e_c) z_g &= z_h^{-1} h(f_c) z_h \\ z_h z_g^{-1} g(e_c) z_g z_g^{-1} &= z_h z_h^{-1} h(f_c) z_h z_g^{-1} \\ c(g,h) g(e_c) &= h(f_c) c(g,h). \end{aligned}$$

□

Definice. Nechť (g, h) je dvojice homomorfismů, která má kritický převis. Potom prefixově minimální dvojici (e, f) splňující

$$c(g,h) g(e) = h(f) c(g,h)$$

nazveme *blok* homomorfismů (g, h). Předchozí lemma tedy říká, že existují nejvýš dva bloky páru (g, h), navíc pokud existují oba, liší se první písmena jejich prvních resp. druhých slov.

Bloky nemarkovaných homomorfismů nejsou „obdélníkové“, jestliže je splněno $c(g, h) \neq \epsilon$. Ovšem začátek ani konec řešení nemohou být „vykousnuté“, první a poslední blok musí mít jiný tvar. Řešení tedy bude složené z jednoznačného začátku, tzv. počátečního bloku, poté z posloupnosti střídajících se písmenných bloků, které již známe, a ukončeno bude nějakým koncovým blokem. Počáteční a koncové bloky dorovnávají právě „vykousnutí“ způsobené tvarem vnitřních bloků.

Lemma 2.13. *Nechť máme dána dvě slova p a q a takové páry (u_1, v_1), (u_2, v_2), že u_1 a u_2 nejsou porovnatelná ani v_1 a v_2 nejsou porovnatelná a $pg(u_i)z_g \bowtie qh(v_i)z_h$ pro $i = 1, 2$. Pak $pg(u_1 \wedge u_2)z_g = qh(v_1 \wedge v_2)z_h$.*

Důkaz. Označme w_i delší ze slov $pg(u_i)z_g$ a $qh(v_i)z_h$ pro $i = 1, 2$. Protože u_1 a u_2 nejsou porovnatelná, není porovnatelné ani $pg(u_1)z_g$ s $pg(u_2)z_g$ podle Důsledku 2.11. Z nerovností $pg(u_1)z_g \leq w_1$ a $pg(u_2)z_g \leq w_2$ plyne

$$w_1 \wedge w_2 = pg(u_1)z_g \wedge pg(u_2)z_g,$$

$$w_1 \wedge w_2 = qh(v_1)z_h \wedge qh(v_2)z_h,$$

druhou rovnost obdržíme stejným způsobem, jen pro druhý homomorfismus. Pak tedy

$$pg(u_1)z_g \wedge pg(u_2)z_g = qh(v_1)z_h \wedge qh(v_2)z_h,$$

důkaz již plyne z Důsledku 2.11. □

Lemma 2.14. *Nechť máme dána dvě slova p a q. Pak existuje nejvýš jedna prefixově minimální dvojice slov (u, v) splňující*

$$pg(u)z_g = qh(v)z_h.$$

Pokud taková existuje, má (g, h) kritický převis a $pg(u) = qh(v)c(g, h)$.

Důkaz. Předpokládejme, že rovnost $pg(u_i)z_g = qh(v_i)z_h$ splňují dvě různé dvojice slov (u_1, v_1) a (u_2, v_2). Předpokládejme nejprve, že u_1 a u_2 jsou porovnatelná, bez újmy na obecnosti nechť $u_1 \leq u_2$. Pak podle Lemmatu 2.13 bude splněno $pg(u_1)z_g \leq pg(u_2)z_g$ a tedy také $qh(v_1)z_h \leq qh(v_2)z_h$ a $v_1 \leq v_2$. To je však spor s minimalitou páru (u_2, v_2). Nutně tedy u_1 není porovnatelné s u_2, obdobně lze dokázat, že nejsou porovnatelná slova v_1, v_2.
Podle Lemmatu 2.13 máme $pg(u_1 \wedge u_2)z_g = qh(v_1 \wedge v_2)z_h$, což je ovšem opět spor s minimalitou obou párů.
Rovnost $pg(u) = qh(v)c(g, h)$ pak plyne okamžitě z definice kritického převisu. □

Definice. Mějme $I = ((p_0, q_0), g, h, (s_0, t_0))$ instanci GPCP. Dvojici (p_1, q_1) nazveme *počáteční blok instance* I, pokud jsou p_1 a q_1 prefixově porovnatelná a je to minimální pár splňující $p_0 g(p_1) = q_0 h(q_1)c(g, h)$. Dvojici slov (u, v) nazveme jeho *koncovým blokem*, jestliže jsou u a v sufixově porovnatelná a tvoří sufixově minimální pár takový, že $c(g, h)g(u)s_0 = h(v)t_0$.

Z Lemmatu 2.14 plyne, že počáteční blok je definovaný jednoznačně. Koncových bloků ovšem může být víc, jak ukazuje následující příklad.

Příklad 7. Mějme instanci (g, h), kde homomorfismy g, h jsou dané vztahy

$$\begin{aligned} g(a) &= ab, \quad & g(b) &= b, \\ h(a) &= ba, \quad & h(b) &= bba \end{aligned}$$

a $(s_1, s_2) = (aa, a)$. V tomto případě máme $z_g = \epsilon$, $z_h = b$, koncový blok (u, v) tedy má splňovat rovnost $bg(u)aa = h(v)a$. Vidíme, že dvojice (b, b) je koncový blok této instance, stejně tak ale i páry (ba, ba) a (baa, baa).

Definice. Pokud existují oba bloky páru (g, h), definujeme *následníka* jako dvojici homomorfismů (g_1, h_1) určenou takto:

$$\begin{aligned} g_1(a) &= e_a \quad & g_1(b) &= e_b \\ h_1(a) &= f_a \quad & h_1(b) &= f_b \end{aligned}$$

Následníkem instance $I = ((p_0, q_0), g, h, (s_0, t_0))$ nazveme každou instanci I', která je ve tvaru $I' = ((p_1, q_1), g_1, h_1, (s_1, t_1))$, kde (g_1, h_1) je následník (g, h), (p_1, q_1) je počáteční blok I a (s_1, t_1) je některý z jeho koncových bloků.

Zavedení pojmu následníka je motivováno *rozkladem řešení w na bloky*, tedy posloupností $((u_i, v_i))_{i=0}^k$, že $w = u_0u_1 \dots u_k = v_0v_1 \dots v_k$, kde (u_0, v_0) je počáteční blok, (u_k, v_k) koncový blok a ostatní páry (u_i, v_i) jsou bloky.
Z Důsledku 2.2 vyplývá, že g_1 i h_1 jsou markované homomorfismy. Takže i pokud původní homomorfismy markované nejsou, změní se to již v první generaci následníků.

Je minimálně nepříjemné, že následníků může být víc - koncový blok totiž není daný jednoznačně. Posloupnost následníků tedy tvoří orientovaný strom, tzv. *redukční strom*, v němž uzly představují instance GPCP a hrany vedou od instance k jejím následníkům. Výpočet následníka je polynomiální, aby byl polynomiální i celý rozhodovací problém, musíme dokázat omezit velikost redukčního stromu. Z kapitoly o markovaných homomorfismech víme, že výška stromu je, až na pár

speciálních případů popsaných v Tvrzení 2.9, omezená (sufixová složitost klesá). Potřebujeme však ještě výrazně omezit šířku stromu, pokud bychom museli projít u každé instance více větví s následníky, řešení by bylo exponenciální. Pokusíme se tedy všechny možné následníky reprezentovat jediným (stejně jako tomu bylo u markovaných homomorfismů), aby nedocházelo k větvení.

Lemma 2.15. *Nechť má dvojice homomorfismů (g, h) následníka (g_1, h_1).*
1. Nechť w je řešení instance $I = ((p, q), g, h, (s, t))$, které je rozložitelné na bloky. Potom existuje následník $I_1 = ((p_1, q_1), g_1, h_1, (s_1, t_1))$ a jeho řešení w_1 takové, že

$$w = p_1 g_1(w_1) s_1 = q_1 h_1(w_1) t_1.$$

2. Naopak je-li w_1 řešením $I_1 = ((p_1, q_1), g_1, h_1, (s_1, t_1))$, následníka I, pak

$$w = p_1 g_1(w_1) s_1 = q_1 h_1(w_1) t_1$$

je řešením instance I rozložitelné na bloky.

Důkaz. 1. Označme rozklad w na bloky jako $(u_0, v_0), (u_1, v_1), \ldots, (u_k, v_k)$. Následník je pak tvaru $I_1 = ((u_0, v_0), g_1, h_1, (u_k, v_k))$. Označíme bloky (g, h) jako dvojice (e_a, f_a) a (e_b, f_b) a definujeme $w_1 = c_1 c_2 \ldots c_{k-1}$, kde

$$c_i = \begin{cases} a, & \text{je-li } (u_i, v_i) = (e_a, f_a), \\ b, & \text{je-li } (u_i, v_i) = (e_b, f_b). \end{cases}$$

Potom $w = u_0 g_1(w_1) u_k = v_0 h_1(w_1) v_k$.
2. Jednotlivá písmena řešení w_1 označíme jako $w_1 = l_1 l_2 \ldots l_d$. Z definice bloků a následníků vidíme, že

$$c(g, h) g(g_1(w_1)) = h(h_1(w_1)) c(g, h)$$

a z definice počátečního a koncových bloků dostaneme

$$p_0 g(p_1 g_1(w_1) s_1) s_0 = q_0 g(q_1 h_1(w_1) t_1) t_0.$$

Proto $w = p_1 g_1(w_1) s_1 = q_1 h_1(w_1) t_1$ je řešení I a jeho rozklad můžeme zapsat jako

$$(p_1, q_1), (g_1(l_1), h_1(l_1)), (g_1(l_2), h_1(l_2)), \ldots, (g_1(l_d), h_1(l_d)), (s_1, t_1).$$

□

Bude nutné ještě důkladněji prozkoumat koncové bloky, poslouží nám k tomu mimo jiné zrcadlové obrazy homomorfismů. Všimněme si, že u koncového bloku (u, v) je pro následníka důležitý hlavně přesah vu^{-1}, což je prvek A^*, pokud $u \leq_s v$, nebo prvek $(A^{-1})^*$ v opačném případě. Nemusíme u následníka jako závěrečná slova brát celá slova u, v, stačí uvažovat jejich přesah. Potom přesah koncových slov následníka (s_1, t_1) bude roven (ϵ, vu^{-1}), resp. (uv^{-1}, ϵ). Zavedeme značení

$$\mathbf{e}(u, v) = \begin{cases} (\epsilon, vu^{-1}), & \text{je-li } vu^{-1} \in A^*, \\ (uv^{-1}, \epsilon), & \text{je-li } vu^{-1} \in (A^{-1})^*. \end{cases}$$

Platí $\mathbf{e}(uz, vz) = \mathbf{e}(uz', vz')$. Ačkoliv koncových bloků může být obecně víc, domníváme se, že jsou maximálně dvě různé hodnoty $\mathbf{e}(u, v)$.

Definice. Existují-li oba bloky (g, h), označíme r (resp. q) největší společný sufix slov $e_a e_b$ a $e_b e_a$ (resp. $f_a f_b$ a $f_b f_a$).

Tato slova jsou obdobou z_g a z_h, ale pro reversní instanci. Při hledání koncových bloků nám budou velmi užitečná, jak později uvidíme. Následující Lemma nám přesně udává, jaký je vztah mezi bloky instance a bloky její reversní instance, mimo jiné vidíme, že buď mají následníka homomorfismy obou těchto instancí, nebo neexistují oba bloky ani pro jednu z nich.

Lemma 2.16. *Nechť (g_1, h_1) je následník (g, h). Pak homomorfismy $(\overline{g}, \overline{h})$ mají následníka (g_1', h_1'), a platí*

$$g_1'(a) = \overline{r e_a r^{-1}}, \quad g_1'(b) = \overline{r e_b r^{-1}}$$
$$h_1'(a) = \overline{q f_a q^{-1}}, \quad h_1'(b) = \overline{q f_b q^{-1}}.$$

Navíc $g(r) = \overline{c(\overline{g}, \overline{h})} h(q) c(g, h)$.

Důkaz. Protože je $\mathrm{pref}_1(e_a) \neq \mathrm{pref}_1(e_b)$, neplatí $e_a e_b = e_b e_a$, a tedy r musí být vlastní sufix slov $e_a e_b$, $e_b e_a$. Také q musí být vlastní sufix $f_a f_b$ i $f_b f_a$. Zřejmě $\overline{r} = z_{\overline{g_1}}$ a $\overline{q} = z_{\overline{h_1}}$ podle jejich definic a

$$g_1'(c) = z_{\overline{g_1}}^{-1} \overline{g_1}(c) z_{\overline{g_1}}, \quad h_1'(c) = z_{\overline{h_1}}^{-1} \overline{h_1}(c) z_{\overline{h_1}},$$

kde $c \in \{a, b\}$. To ale podle definice znamená, že g_1' je markovanou verzí homomorfismu $\overline{g_1}$ a h_1' je markovanou verzí $\overline{h_1}$, proto jsou tyto homomorfismy dobře definované. Zbývá ještě dokázat, že (g_1', h_1') je následník (g, h). Musíme tedy ověřit, že

$$c(\overline{g}, \overline{h}) \overline{g}(\overline{r^{-1}}\ \overline{e_a}\ \overline{r}) = \overline{h}(\overline{q^{-1}}\ \overline{f_a}\ \overline{q}) c(\overline{g}, \overline{h}),$$
$$c(\overline{g}, \overline{h}) \overline{g}(\overline{r^{-1}}\ \overline{e_b}\ \overline{r}) = \overline{h}(\overline{q^{-1}}\ \overline{f_b}\ \overline{q}) c(\overline{g}, \overline{h})$$

a páry $(\overline{r^{-1}}\ \overline{e_c}\ \overline{r}, \overline{q^{-1}}\ \overline{f_c}\ \overline{q})$ jsou minimální, které to splňují pro $c = a, b$. Protože (e_c, f_c) jsou bloky, musí platit pro každé $i \in \mathbb{N}$:

$$\overline{g}(\overline{e_a}\ \overline{e_b})^i \overline{c(g, h)} = \overline{c(g, h)} \overline{h}(\overline{f_a}\ \overline{f_b})^i,$$
$$\overline{g}(\overline{e_b}\ \overline{e_a})^i \overline{c(g, h)} = \overline{c(g, h)} \overline{h}(\overline{f_b}\ \overline{f_a})^i.$$

Podle Lemmatu 2.13 a definice r a q vidíme, že kritický převis $c(\overline{g}, \overline{h})$ existuje, je roven $z_{\overline{h}} z_{\overline{g}}^{-1}$ a platí:

$$\overline{g}(\overline{r}) = \overline{c(g, h)}\ \overline{h}(\overline{q}) c(\overline{g}, \overline{h}).$$

Pomocí této rovnosti a faktu, že (e_a, f_a) je blok, obdržíme:

$$\overline{g}(\overline{e_a}\ \overline{r}) = \overline{g}(\overline{e_a}) \overline{c(g, h)} \overline{h}(\overline{q}) c(\overline{g}, \overline{h}) = \overline{c(g, h)}\ \overline{h}(\overline{f_a}\ \overline{q}) c(\overline{g}, \overline{h}),$$

a opětovným použitím stejné rovnosti tedy bude

$$\overline{g}(\overline{r^{-1}}\ \overline{e_a}\ \overline{r}) = c(\overline{g}, \overline{h})^{-1}\ \overline{h}(\overline{q^{-1}}\ \overline{f_a}\ \overline{q}) c(\overline{g}, \overline{h}).$$

Tímto je splněna požadovaná rovnost, druhou obdržíme obdobně.
Dvojice $(\overline{r^{-1}}\ \overline{e_c}\ \overline{r}, \overline{q^{-1}}\ \overline{f_c}\ \overline{q})$ jsou zřetězením bloků instance $(\overline{g}, \overline{h})$, zbývá ukázat jejich minimalitu. Pokud tedy tyto dvojice minimální nejsou, má pár $(\overline{g}, \overline{h})$ kratší bloky než (g, h), tedy součet délek všech čtyř slov z těchto bloků je ostře menší než součet délek slov z bloků (g, h). Ale g, h jsou zrcadlové obrazy $\overline{g}$, $\overline{h}$ a mohli bychom pomocí první části tohoto důkazu získat kratší bloky instance (g, h), což je spor s definicí bloků. □

Jak nyní dokážeme, nemohou být slova r i q rovna některému ze slov bloků. Předchozí i následující Lemma využijeme v dalších důkazech tvrzení o koncových blocích. Poté budeme i vědět, jak koncové bloky jednoduše zkonstruovat.

Lemma 2.17. *Platí: r je vlastní sufix e_a i e_b nebo q je vlastní sufix f_a i f_b.*

Důkaz. Pro spor předpokládejme opak, tedy že neplatí ani jedna část tvrzení. Pak jsou e_a a e_b sufixově porovnatelná, stejně tak i f_a a f_b. Nechť bez újmy na obecnosti $e_a \leq_s e_b$, pak $|g(e_a)| = |h(f_a)| \leq |g(e_b)| = |h(f_b)|$, takže musí být f_a sufixem f_b. To je ale spor s minimalitou bloku (e_b, f_b), neboť dvojice $(e_b e_a^{-1}, f_b f_a^{-1})$ je menší a také splňuje požadovanou rovnost

$$c(g,h)g(e_b e_a^{-1}) = c(g,h)g(e_b)g(e_a^{-1}) = h(f_b)h(f_a^{-1})c(g,h) = h(f_b f_a^{-1})c(g,h).$$

□

Tvrzení 2.18. *Nechť mají I i $\overline{I}$ následníka. Pak počet koncových bloků instance I je maximálně* $\max\{|e_a e_b|, |f_a f_b|\} + 1$. *Navíc pro jakýkoliv koncový blok (u, v) platí*

$$ru = \overline{g_1'(z)}b_1, \qquad qv = \overline{h_1'(z)}b_2,$$

kde $(\overline{b_1}, \overline{b_2})$ je počáteční blok $\overline{I}$ a slovo z je minimální takové, že

$$r \leq \overline{g_1'(z)}b_1, \qquad q \leq \overline{h_1'(z)}b_2.$$

Důkaz. Vezměmě koncový blok (u, v). Pak podle Lemmatu 2.16 platí:

$$g(ru)s_1 = \overline{c(\overline{g}, \overline{h})}h(qv)s_2$$

proto tedy máme

$$ru = \overline{g_1'(z)}b_1, \quad qv = \overline{h_1'(z)}b_2,$$

pro nějaké z. Užitím Lemmatu 2.16 tedy dostaneme

$$u = r^{-1}\overline{g_1'(z)}b_1 = g_1(\overline{z})r^{-1}b_1, \quad v = q^{-1}\overline{h_1'(z)}b_2 = h_1(\overline{z})q^{-1}b_2.$$

Protože je koncový blok (u, v) sufixově minimální, musí být z minimální slovo splňující obě nerovnosti v závěru tvrzení.
Je-li r sufix obou slov e_a i e_b a také q je sufix f_a i f_b, je slovo z buď prázdné nebo má délku jedna. Předpokládejme tedy například, že e_a a e_b jsou sufixově porovnatelná. Označme k nejmenší číslo takové, že r je sufix e_a^k a podobně m nejmenší takové, že r je sufix e_b^m. Navíc aby z splňovalo uvedené nerovnosti, musí být $g_1'(z)$ délky alespoň $|e_a e_b|$. Proto buď $m \leq 2$ v případě $|e_a| \leq |e_b|$, nebo $k \leq 2$ ve druhém případě. Je vidět, že

$$z \in \{a^i b | i = 1, 2, \cdots, k-1\} \cup \{b^j a | j = 1, 2, \cdots, m-1\} \cup \{a^k, b^m\}.$$

Protože je splněno $k, m \leq |r|$, počet koncových bloků je nejvýše $|r| + 2$. Stejným způsobem obdržíme hranici $|q| + 2$, jsou-li sufixově porovnatelná f_a a f_b. Tímto dostáváme uvedenou mez $\max\{|e_a e_b|, |f_a f_b|\} + 1$. □

Tvrzení 2.19. *Nechť mají I i $\overline{I}$ následníka. Nechť (u_1, v_1), (u_2, v_2) jsou dva různé koncové bloky I takové, že ru_i a qv_i jsou sufixově porovnatelná pro $i = 1, 2$. Pak existuje slovo τ splňující*

$$g(\tau)s_1 = \overline{c(\overline{g}, \overline{h})}h(\tau)s_2, \tag{2.4}$$

a (τ, τ) je sufixem každého (ru, qv), kde (u, v) je koncový blok se sufixově porovnatelnými ru a qv.

Důkaz. Nechť jsou (u_i, v_i) všechny koncové bloky pro $i = 1, 2, \cdots, k$ takové, že všechny dvojice (ru_i, qv_i) jsou sufixově porovnatelné. Označme (g'_1, h'_1) následníka $(\overline{g}, \overline{h})$. Z Tvrzení 2.18 plynou rovnosti

$$ru_i = \overline{g'_1(z_i)}b_1, \quad qv_i = \overline{h'_1(z_i)}b_2,$$

kde $(\overline{b_1}, \overline{b_2})$ je počáteční blok instance $\overline{I}$ a z_i, $i = 1, 2, \cdots, k$, jsou prefixově minimální, tedy žádné není prefixem jiného. Největší společný prefix všech z_i označíme z a bez újmy na obecnosti budeme předpokládat, že $z = z_1 \wedge z_2$. Protože slova z_1 a z_2 nejsou porovnatelná, musí být $z_1 = zz'_1$ a $z_2 = zz'_2$ pro z'_1, z'_2 neprázdná a navíc $\mathrm{pref}_1(z'_1) \neq \mathrm{pref}_1(z'_2)$. Ze sufixové porovnatelnosti slov ru_1 a ru_2 a také slov qv_1, qv_2 a díky markovanosti následníka (g'_1, h'_1) dostáváme

$$\overline{g'_1(z)}b_1 = ru_1 \wedge_s ru_2 = qv_1 \wedge_s qv_2 = \overline{h'_1(z)}b_2.$$

Pokud definujeme slovo τ jako kterékoliv slovo z uvedené rovnosti, bude pro něj splněna požadovaná rovnice ze závěru tvrzení. Navíc je slovo τ sufixem všech slov ru_i i qv_i, což plyne z volby slova z. □

Nyní konečně můžeme dokázat, jak lze následníky redukovat do jednoho a jaký vliv má tato redukce na řešení zadané instance. Uvidíme, že některá krátká řešení bychom mohli redukcí ztratit, proto při přechodu od instance k jejímu následníkovi budeme muset zkontrolovat, zda neexistuje nějaké krátké řešení.

Definice. Řekneme, že řešení w instance I je *dlouhé*, právě když existuje τ splňující (2.4) a $w = p'_1 w_1 \tau = p'_2 w_2 \tau$, pro neprázdná slova w_1, w_2 a (p'_1, p'_2) počáteční blok I. Pokud řešení není dlouhé, řekneme, že je *krátké*.

Protože τ je sufix ru i qv pro každý koncový blok (u, v), který splňuje $ru \bowtie_s qv$, má krátké řešení délku nejvýš $\max(|p'_1 e_a e_b u|, |p'_2 f_a f_b v|)$, kde maximum bereme přes všechny koncové bloky splňující výše uvedenou podmínku. Řešení, která mají délku nejvýš $\max(|p'_1 e_a e_b u|, |p'_2 f_a f_b v|)$ nazveme *nedlouhá řešení*. Vidíme, že každé krátké řešení je zároveň nedlouhé, ovšem některé dlouhé řešení může být zároveň nedlouhé.
Následníci, jejichž koncové bloky (u, v) nesplňují $ru \bowtie_s qv$, mohou poskytnout řešení nejvýš délky $\max(|p'_1 ru|, |p'_2 qv|)$, protože (r, q) je sufixově porovnatelné s oběma bloky.

Tvrzení 2.20. *Nechť mají I i $\overline{I}$ následníka. Nechť (u_1, v_1), (u_2, v_2) jsou dva různé koncové bloky I takové, že ru_i a qv_i jsou sufixově porovnatelná pro $i = 1, 2$. Označme (g_1, h_1) následníka (g, h) a předpokládejme, že i (g_1, h_1) má následníka. Potom jsou q a r porovnatelná. Definujme instanci $J = ((p'_1, p'_2), g_1, h_1, \mathbf{e}(\epsilon, q^{-1}r))$, kde (p'_1, p'_2) je počáteční blok I. Pak*
1) Má-li I dlouhé řešení, má J neprázdné řešení.
2) Pokud má J neprázdné řešení, má I řešení.

Důkaz. Podle Lemmatu 2.16 mají i homomorfismy $(\overline{g_1}, \overline{h_1})$ následníka, a mají tedy kritický převis. To znamená, že $\overline{q}$ a $\overline{r}$ jsou sufixově porovnatelná, takže r a q jsou prefixově porovnatelná.
1) Mějme $w = p'_1 w_1 \tau = p'_2 w_2 \tau$ dlouhé řešení instance I. Z definice počátečního bloku a Tvrzení 2.19 víme, že $c(g,h)g(w_1)\overline{c(\overline{g},\overline{h})} = h(w_2)$. Díky Lemmatu 2.16 je tedy $c(g,h)g(w_1 r) = h(w_2 q)c(g,h)$. Potom tedy existuje slovo w', které splňuje $g_1(w') = w_1 r$, $h_1(w') = w_2 q$. To znamená, že

$$p'_1 w_1 \tau = p'_2 w_2 \tau$$

$$p'_1 g_1(w') r^{-1} \tau = p'_2 h_1(w') q^{-1} \tau,$$

w' je tedy řešení instance J. Protože $g(w') = w_1 r$ a w_1 je neprázdné, je i w' nenulové řešení.
2) Nechť w' je neprázdné řešení J, tedy

$$p'_1 g_1(w') = p'_2 h_1(w') q^{-1} r.$$

Podle Lemmatu 2.17 je $p'_1 g_1(w') r^{-1} = p'_2 h_1(w') q^{-1} \in A^+$. Protože

$$p_1 g(p'_1 g_1(w')) = p_2 h(p'_2 h_1(w')) c(g,h),$$

dávají Lemma 2.16 a Tvrzení 2.19

$$p_1 g(p'_1 g_1(w') r^{-1} \tau) s_1 = p_2 h(p'_2 h_1(w') q^{-1} \tau) s_2,$$

a tedy $p'_1 g_1(w') r^{-1} \tau = p'_2 h_1(w') q^{-1} \tau$ je řešením prku I. □

Toto tvrzení nám tedy dává návod na to, jak zredukovat následníky do jediného. Jestliže již neexistuje následník tohoto následníka a Tvrzení tedy nelze použít, dostali jsme se již na poslední úroveň redukčního stromu a můžeme si tedy dovolit vyšetřit všechny následníky zvlášť. Další větvení totiž již nenastane a polynomialita tak nebude narušena.

3. Kandidáti na řešení

Řešení GPCP samozřejmě lze sestavovat postupně přidáváním písmen k již vzniklému začátku řešení *(předřešení)* (w_g, w_h) podle převisů jeho obrazů, jak uvádí Důsledek 2.11. Takto lze předřešení prodlužovat pouze za předpokladů, že lze o pokračování rozhodnout (tj. převis není kritický) a slova $p_1g(w_g)z_g$ a $p_2h(w_h)z_h$ jsou porovnatelná. Jestliže narazíme na kritický převis, lze předřešení prodloužit podle delšího z obou slov nalezeného předřešení. Jsou-li však tato slova stejná (tzv. *speciální situace*), musíme vyzkoušet obě možné varianty pokračování. Dostaneme-li se k předřešení (w_g, w_h), pro které slova $p_1g(w_g)z_g$ a $p_2h(w_h)z_h$ nejsou porovnatelná, stále může toto předřešení vést k řešení. Mohou totiž existovat slova, která mají obraz kratší než nutný začátek z_g, resp. z_h (tzv. finální slova).
Této konstrukce bychom chtěli využít při hledání řešení neperiodické instance, která nemá následníka (tj. chybí některý vnitřní blok) nebo neexistuje její počáteční blok, anebo známe-li omezení na délku řešení. Při neexistenci vnitřního nebo počátečního bloku musíme ještě odhalit, jak daleko je třeba při hledání řešení jít. Předpokládejme tedy, že máme instanci, které chybí počáteční nebo některý písmenný blok. Také může hledání maximálních dostatečných kandidátů algoritmus ukončit rychleji než například přechod k následnické instanci.

3.1 Dostateční kandidáti

Definujeme několik pojmů, které nám pomohou formulovat tuto část řešení přesně. Tyto pojmy jsou si však dost podobné. Nejprve definujeme dostatečného kandidáta jako takové slovo, ze kterého můžeme získat řešení pouze přidáváním finálních slov k jeho prefixům a přitom je toto řešení jistým způsobem minimální. Nejprve ale základní definice:

Definice. Pro danou instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$ definujeme *předřešení* jako každou dvojici slov (w_g, w_h) takovou, že platí:

- $w_g \bowtie w_h$ a
- $p_1g(w_g)z_g \bowtie p_2h(w_h)z_h$.

Navíc o předřešení (w_g, w_h) řekneme, že tvoří *speciální situaci*, pokud jsou v obou podmínkách rovnosti, to znamená $w_g = w_h$ a $p_1g(w_g)z_g = p_2h(w_h)z_h$.
O předřešení (w'_g, w'_h) řekneme, že *obsahuje speciální situaci*, pokud nějaký jeho prefix $(x, y) \leq (w_g, w_h)$ tvoří speciální situaci. Obdobně o slově w řekneme, že *obsahuje speciální situaci*, pokud ji obsahuje dvojice (w, w).

Jak bylo naznačeno v úvodu, mohou existovat slova, jejichž přidání k předřešení (w_g, w_h) neporuší podmínku porovnatelnosti obrazů samotného řešení, tedy $p_1g(w_g) \bowtie p_2h(w_h)$, přestože již poté nebude splněno $p_1g(w_g)z_g \bowtie p_2h(w_h)z_h$. Takové prodloužení však stále může vést k řešení. Tato slova nazýváme *finální*.

Definice. Množinu $F = \{u \in \{a, b\}^* : |g(u)| < |z_g|$ nebo $|h(u)| < |z_h|\}$ nazveme *množinou finálních slov instance* $I = ((p_1, p_2), g, h, (s_1, s_2))$.

Protože $z_g = g(ab) \wedge g(ba)$ a $z_h = h(ab) \wedge h(ba)$, obsahuje množina F pouze slova, která jsou mocninami jediného písmene. Prázdné slovo je také finálním slovem. Podle Lemmatu 3.1 je finálních slov dané instance celkem méně než $|z_g| + |z_h|$.

Lemma 3.1. *Je-li g neperiodický (tedy i nenulující) homomorfismus, pak množina $F_g = \{u \in \{a,b\}^* : |g(u)| < |z_g|, u \neq \epsilon\}$ obsahuje méně než $|z_g|$ prvků.*

Důkaz. Bez újmy na obecnosti můžeme předpokládat, že $|g(a)| \leq |g(b)|$. Víme, že každé $u \in F_g$ je tvaru $u = a^i$ nebo $u = b^j$, $i, j \in \mathbb{N}$.
1. Pokud $|g(a)| \geq 2$, tedy i $|g(b)| \geq 2$, může mít slovo $u = a^i$ obraz kratší než $|z_g|$, pouze pokud $i < \frac{|z_g|}{2}$. Obdobně může platit $|g(b^j)| < |z_g|$ pouze pro exponent $j < \frac{|z_g|}{2}$. Celkově je tedy slov v množině F_g méně než $|z_g|$.
2. Je-li $|g(a)| = 1$, je slov tvaru $u = a^i$ v množině F_g nejvýš $|z_g| - 1$. Protože $z_g < g(a)g(b)$, je obraz slova b dlouhý alespoň $|z_g|$, takže množina F_g neobsahuje slova tvaru b^j. Velikost této množiny je tedy nejvýš $|z_g| - 1$. □

Definice. Mějme w řešení instance I. Slovo w_m nazveme *kmenem řešení* w, právě když je w_m nejdelší prefix w splňující $p_1 g(w_m) z_g \bowtie p_2 h(w_m) z_h$.

Máme-li tedy instanci I a kmen w_m nějakého jejího řešení w, musí být každá dvojice (w', w') předřešením této instance pro $w' \leq w_m$.
Kdyby nějaké řešení w obsahovalo dva prefixy, v nichž nastává speciální situace, můžeme toto řešení zkrátit:
Nechť existují neprázdná slova w_1, w_2 taková, že $w = w_1 w_2 w_3$ a splňující

$$p_1 g(w_1) c(g,h) = p_2 h(w_1) \quad \text{a} \quad p_1 g(w_1 w_2) c(g,h) = p_2 h(w_1 w_2).$$

Protože jsou shodné převisy obrazů w_1 a $w_1 w_2$, musí být $|g(w_2)| = |h(w_2)|$, tedy

$$g(w_2) c(g,h) = c(g,h) h(w_2).$$

Z toho plyne, že

$$p_1 g(w_1 w_3) \bowtie p_2 h(w_1 w_3)$$

a $p_1 g(w_1 w_3)$ a $p_2 h(w_1 w_3)$ tvoří stejný převis jako $p_1 g(w)$ s $p_2 h(w)$, je tedy také slovo $w_1 w_3$ řešením.

Definice. Kmen w_m řešení w nazveme *dostatečným kandidátem na řešení instance* I, pokud
1. Neobsahuje dvě speciální situace, tj. nelze w_m rozdělit na $w_m = w_1 w_2 w_3$, w_1, w_2 neprázdná, aby platilo

$$p_1 g(w_1) c(g,h) = p_2 h(w_1) \quad \text{a} \quad p_1 g(w_1 w_2) c(g,h) = p_2 h(w_1 w_2)$$

2. Pro každý kmen w'_m nějakého řešení w' platí

$$w'_m \leq w_m \implies w_m = w'_m.$$

Dostateční kandidáti jsou tedy prefixově minimální kmeny s nejvýš jednou speciální situací. Pokud instance nemá řešení, neexistuje žádný dostatečný kandidát této instance, protože neexistují žádné kmeny řešení. Pokud však řešení existuje, ani jedna z podmínek v definici nezajišťuje jednoznačnost dostatečných kandidátů, může jich být tedy víc. Následující tvrzení však ukáže, že mohou být nejvýš dva.

Tvrzení 3.2. *Pro danou instanci I existují nejvýš dva dostateční kandidáti na řešení této instance.*

Důkaz. Mějme tři různé dostatečné kandidáty w_1, w_2, w_3 a označme jejich společné prefixy

$$\alpha_{1,2} = w_1 \wedge w_2,$$

$$\alpha_{2,3} = w_2 \wedge w_3,$$

$$\alpha_{3,1} = w_1 \wedge w_3.$$

Slova $\alpha_{i,j}$, $\alpha_{j,k}$ jsou každé prefixem slova w_j pro $\{i,j,k\} = \{1,2,3\}$, takže musí být po dvou porovnatelná. Předpokládejme bez újmy na obecnosti $\alpha_{1,2} \leq \alpha_{2,3} \leq \alpha_{3,1}$. Kdyby $w_i = \alpha_{i,j}$, bude $w_i \leq w_j$ pro $(i,j) \in \{(1,2),(2,3),(3,1)\}$. Protože jsou však w_i i w_j dostateční kandidáti, musí být $w_i = w_j$, což je spor s předpokladem, že jsou kandidáti různí. Proto $\alpha_{i,j} < w_i$ a $\alpha_{i,j} < w_j$. Takže $\alpha_{1,2} < w_1, w_2, w_3$.
Navíc pokud by bylo $\alpha_{1,2} = \alpha_{2,3} = \alpha_{3,1}$, na $(\alpha_{1,2}+1)$-ních místech slov w_1, w_2, w_3 musí být různá písmena, což nelze.
Víme tedy, že $\alpha_{1,2} < \alpha_{3,1}$. Navíc kandidáti w_1, w_2 mají na $(\alpha_{1,2}+1)$-ních místech různá písmena, takže

$$p_1 g(\alpha_{1,2}c) z_g \bowtie p_2 h(\alpha_{1,2}c) z_h$$

pro $c = a$ i $c = b$ - tato porovnatelnost je splněna pro každé podslovo dostatečného kandidáta díky Lemmatu 2.10. To ale může nastat pouze pokud je splněno $p_1 g(\alpha_{1,2}) z_g = p_2 h(\alpha_{1,2}) z_h$, máme tedy speciální situaci. Obdobně obdržíme speciální situaci i ve druhém slově $\alpha_{3,1}$, což je ale spor s tím, že slovo w_3 je dostatečným kandidátem - obsahovalo by dvě speciální situace. □

3.2 Maximální kandidáti na řešení

Zavedeme ještě pojem *maximálního kandidáta*, který sice může být delší, než dostatečný kandidát, ale přesto nebude dlouhý příliš. Tento pojem navíc využijeme v implementaci algoritmu, jelikož k jeho nalezení nemusíme vědět, jestli má instance řešení. Maximální kandidáty tedy budeme umět hledat algoritmicky, v algoritmu ve skutečnosti budeme pracovat maximálně se dvěma z nich, s tzv. *maximálními dostatečnými kandidáty.* Maximální kandidáti tedy budou prodloužením dostatečných kandidátů a maximální dostateční kandidáti budou ti nejkratší z maximálních kandidátů, ze kterých lze nalézt řešení, pokud existuje. Stačí vyzkoušet, jestli nějaký prefix maximálního dostatečného kandidáta spolu s jedním z finálních slov tvoří řešení dané instance.

Definice. Slovo w' nazveme *maximálním kandidátem na řešení instance I*, právě když platí:

- je-li w' prefixově porovnatelný s kmenem nějakého řešení instance I, pak některý jeho prefix je dostatečným kandidátem na řešení instance I,
- $p_1 g(w') z_g \bowtie p_2 h(w') z_h$,
- w' neobsahuje dvě speciální situace.

Maximálního kandidáta budeme sestrojovat postupně prodlužováním předřešení, jak bylo popsáno na začátku této kapitoly. Pokud se dostaneme ke sporu buď v obrazech daného předřešení nebo dokonce u vzorů, tj. neplatí $w_g \bowtie w_h$, k řešení se můžeme dostat již jen finálním slovem, maximálního kandidáta tedy už máme.
Jestliže rozpor nenastává, musí se některý z převisů vyskytnout podruhé, neboť je převisů obrazů pouze konečně mnoho. Pokud tedy narazíme podruhé na stejný převis α, zjistíme dvojice slov, které k němu vedly, tj. (w'_g, w'_h) a $(w'_g x, w'_h y)$, že $\alpha g(x) = h(y)\alpha$. V této situaci by se mohlo stát, že budeme prodlužovat řešení donekonečna, musíme tedy zjistit, kdy už je přidávání dalších písmen zbytečné.
Vyskytne-li se podruhé shodný převis a žádný převis mezi nimi nebyl kritický, je pokračování vždy jednoznačné, přidávat lze jen dvojici (x, y). Může se také stát, že jeden z převisů mezi dvěma shodnými převisy je kritický. Ten se jistě objeví i v následujícím „bloku“, takže můžeme místo původního bloku (x, y) vzít ten, který leží mezi kritickými převisy. V tomto bloku již není kritický převis obsažen. Maximální kandidát je určen následujícími tvrzeními (podmínky 1. a 2. lze poznat snadno díky Tvrzení 1.3).

Tvrzení 3.3. *Nechť pro nějakou instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$ existují slova w'_g, w'_h a neprázdná primitivní slova x, y, že*
1. $w'_g x^\infty = w'_h y^\infty$,
2. $p_1 g(w'_g x^\infty) = p_2 h(w'_h y^\infty)$,
3. $m, n > 0$ jsou minimální taková, že

$$(p_1 g(w'_g))^{-1} p_2 h(w'_h) = (p_1 g(w'_g x^m))^{-1} p_2 h(w'_h y^n),$$

tedy převisy jsou stejné, a
4. neplatí $p_1 g(w'_g x_1) z_g = p_2 h(w'_g y_1) z_h$ pro žádnou dvojici $(x_1, y_1) \leq (x^m, y^n)$.
Potom existuje k takové, že maximálním kandidátem je slovo $\max(w'_g x^{km}, w'_h y^{kn})$.

Důkaz. Ze třetího předpokladu víme, že $|g(x^m)| = |h(y^n)|$. Protože platí 1. bod a x, y jsou primitivní, podle Tvrzení 1.3 je $|x| = |y|$ a slova x a y jsou konjugovaná. Navíc díky 4. víme, že pokud existuje w kmen řešení, který má prefix w'_g i w'_h, neplatí pro žádné jeho podslovo takové, že $w' \leq w$, $\max(w'_g, w'_h) \leq w'$, vztah $p_1 g(w') z_g = p_2 h(w') z_h$ (speciální situace). Tento kmen tedy bude tvaru

$$w'_h y^{ln} y' = w'_g x^{lm} x'$$

pro $l \geq 2$, $y' \leq y^n$, $x' \leq x^m$. Pokud $w'_h y^{(l-1)n} > w'_g$ a $w'_g x^{(l-1)m} > w'_h$, je kmenem řešení i slovo

$$w'_h y^{(l-1)n} y' = w'_g x^{(l-1)m} x'.$$

Proto stačí volit $k \geq 1$ jako nejmenší takové, že $w'_h y^{(k-1)n} > w'_g$ a $w'_g x^{(k-1)m} > w'_h$. Pak bude slovo $\max(w'_g x^{km}, w'_h y^{kn})$ maximálním kandidátem. □

Tvrzení 3.4. *Nechť pro nějakou instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$ existují slova w'_g, w'_h a neprázdná primitivní slova x, y, že*
1. $w'_g x^\infty = w'_h y^\infty$,
2. $p_1 g(w'_g x^\infty) = p_2 h(w'_h y^\infty)$,
3. m, n jsou minimální nenulová taková, že $p_1 g(w'_g x^m) z_g = p_2 h(w'_h y^n) z_h$,
4. $p_1 g(w'_g) z_g = p_2 h(w'_h) z_h$ a přitom pro žádnou dvojici $(x_1, y_1) \neq (\epsilon, \epsilon)$ podslov (x^m, y^n), $(x_1, y_1) \neq (x^m, y^n)$, není splněna rovnost $p_1 g(w'_g x_1) z_g = p_2 h(w'_g y_1) z_h$.
Pak existuje k takové, že maximálním kandidátem je slovo $max(w'_g x^{km}, w'_h y^{kn})$.

Důkaz. Opět z prvního předpokladu vyplývá a primitivity, že x a y jsou konjugovaná slova, speciálně jsou tedy stejně dlouhá.
1) Pokud $m = n$ buď slova (w'_g, w'_h) tvoří speciální situaci a pak totéž platí i o dvojici $(w'_g x^m, w'_h y^n)$ nebo speciální situace neobsahuje $(w'_g x^{km}, w'_h y^{kn})$ pro žádné k. Navíc je převis předřešení $(w'_g x^{km}, w'_h y^{kn})$ stejný jako převis $(w'_g x^{(k+1)m}, w'_h y^{(k+1)n})$, maximálním dostatečným kandidátem v obou těchto situacích tedy musí být delší ze slov $w'_g x^m$, $w'_h y^n$ zkrácených o poslední písmeno.
2) Mějme tedy $m \neq n$. Speciální situace může nastat pro dvojici $(w'_g x^{jm}, w'_h y^{jn})$, kde $j \geq 0$, pokud

$$w'_g x^{jm} = w'_h y^{jn}. \tag{3.1}$$

Potom tedy srovnáním délek slov v rovnici (3.1) dostaneme:

$$|w'_g| + jm|x| = |w'_h| + jn|y|$$

$$j = \frac{|w'_g| - |w'_h|}{|x|(n-m)}$$

Je tedy zřejmé, že rovnice (3.1) má nejvýš jedno nezáporné řešení. To znamená, že speciální situace nastane nejvýš jedna. Navíc platí

$$||w'_g x^{(l+1)m}| - |w'_h y^{(l+1)n}|| > ||w'_g x^{lm}| - |w'_h y^{ln}||$$

pro každé $l > j$. Zvolíme-li tedy $k > j$ takové, že $||w'_g x^{km}| - |w'_h y^{kn}|| > |z_g|, |z_h|$, bude maximálním kandidátem slovo $\max(w'_g x^{km}, w'_h y^{kn})$. □

Maximálních kandidátů může být obecně mnoho, jsou to totiž všechna slova delší než dostatečný kandidát na řešení, která mají porovnatelné obrazy a neobsahují dvě speciální situace. Ovšem v algoritmu budeme pracovat pouze s těmi maximálními kandidáty, o kterých již s jistotou víme, že nejsou kratší než s nimi porovnatelný dostatečný kandidát, o kterém však ale v tu chvíli ani nevíme, zda existuje. Proto ještě zavedeme pojem *maximální dostatečný kandidát na řešení dané instance.*
Funkce hledající všechny maximální dostatečné kandidáty pro zadanou instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$ tedy bude postupně k již nalezenému předřešení dané instance I přidávat písmena podle převisu obrazů nebo podle pokračování delšího z obou předřešení (v situaci kritického převisu), dokud to lze, tj. dokud nenastane některý z těchto případů:
a) Je porušeno $p_1 g(w_g) z_g \bowtie p_2 g(w_h) z_h$ - to znamená, že nelze přidat slovo w', pro které $|g(w')| \geq |z_g|$ a $|h(w')| \geq |z_h|$. Takže jsme již nalezli maximálního kandidáta - delší ze slov w_g, w_h. Toto slovo pak nazveme *maximálním dostatečným kandidátem na řešení instance I.*
b) Pokud se dostaneme ke kritickému převisu a $w_g = w_h$ (speciální situace). Zde není další pokračování jasné. Pokud tato situace nastala poprvé, spustíme funkci rekurzivně s oběma možnými pokračováními. Pokud ale již k takovému případu došlo předtím, tedy $w'_g = w'_h$ a $p_1 g(w'_g) z_g = p_2 g(w'_h) z_h$ pro nějaký prefix $w'_g = w'_h < w_g = w_h$, lze z řešení, které by obsahovalo w_g, vynechat $(w'_g)^{-1} w_g$. Pak by řešení pokračovalo buď písmenem a nebo b, což jsme vlastně vyřešili již v první speciální situaci. Proto v tomto případě maximálním kandidátem bude slovo $w_g = w_h$ bez posledního písmene a tohoto maximálního kandidáta nazveme *maximálním dostatečným kandidátem na řešení instance I.*

c) Vzniklý převis již jednou o prodloužení rozhodl. Maximální kandidáty v případě donekonečna porovnatelných vzorů i obrazů popisují Tvrzení 3.3, 3.4 a *maximální dostatečný kandidát na řešení I* je slovo $\max(w'_g x^{km}, w'_h y^{kn})$. Jestliže však nemáme vzory či obrazy nekonečně porovnatelné, bude *maximálním dostatečným kandidátem* nejdelší slovo, které ještě nezpůsobuje tento spor. Tuto situaci naznačuje následující příklad.

Příklad 8. Mějme $I = ((p_1, p_2), g, h, (s_1, s_2))$ instanci GPCP, pro kterou jsou $p_1 = ab$, $s_2 = baba$ a slova p_2, s_1 jsou prázdná a

$$\begin{aligned} g(a) &= bababa, & g(b) &= bb, \\ h(a) &= abba, & h(b) &= bababa. \end{aligned}$$

Při sestrojování maximálního dostatečného kandidáta tedy konstruujeme předřešení: Prvním předřešením je (ϵ, a), podle převisu jeho obrazů bude dalším předřešením (a, a), a třetí předřešení má tvar (a, ab). Předřešení $(w'_g, w'_h) = (\epsilon, a)$ a $(w'_g x, w'_h y) = (a, ab)$ vytváří stejný převis a došlo by tedy k nekonečnému přidávání dvojice $(x, y) = (a, b)$. Je jasné, že počáteční blok neexistuje, a že všechna předřešení jsou tvaru (a^k, ab^k). Protože však $a^\infty \neq ab^\infty$, tzn. nejsou porovnatelné oba vzory, bude maximálním dostatečným kandidátem slovo ab, tedy delší slovo z posledního předřešení, kdy ještě jeden vzor byl prefixem druhého. Při hledání řešení víme, že řešením je slovo, které vznikne přidáním finálního slova k nějakému prefixu maximálního dostatečného kandidáta. V tomto případě žádná neprázdná finální slova neexistují a je snadné ověřit, že řešením je slovo a.

Tímto postupem tedy dostaneme jednoho nebo dva maximální kandidáty. Pokud existuje dostatečný kandidát na řešení zadané instance, je prefixem jednoho z nich.
Protože může maximální dostatečný kandidát obsahovat nejvýš jednu speciální situaci a je daný při jakémkoliv končení jednoznačně, mohou být nejvýš dva.

Lemma 3.5. *Mějme I instanci GPCP. Pak platí:*

1. *Instance I má nějakého maximálního dostatečného kandidáta na řešení, právě když $p_1 z_g \bowtie p_2 z_h$.*

2. *Existují nejvýš dva maximální dostateční kandidáti na řešení instance I.*

Důkaz.

1. Plyne ihned z definice maximálního kandidáta.

2. Je zřejmá z předešlého odstavce.

□

4. Nevyřešené případy

V některých případech ovšem neexistují bloky, následníci se nezjednodušují nebo nelze využít postupu při hledání maximálních dostatečných kandidátů. Těchto případů však není mnoho, pro každý z nich ukážeme, jak najít řešení takových instancí.

4.1 Stálá sufixová složitost

Následující část textu řeší případy se stálou sufixovou složitostí. Jedná se pouze o tři různé dvojice balancovaných markovaných homomorfismů (až na záměnu písmen). Nejprve na konkrétních příkladech ukážeme, že řešení lze zjistit snadno ze slov p_1, p_2, s_1, s_2.

Příklad 9. Mějme instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$, kde g i h je identita. Nechť $p_1 \neq \epsilon$, $p_2 = \epsilon$, $s_1 = \epsilon$. Hledejme, jaký tvar musí mít slovo s_2, aby existovalo řešení. Je vidět, že k existenci řešení je nutnou podmínkou, aby slova p_1 a s_2 měla stejnou délku. Pokud existuje slovo u takové, že $p_1 = u_1 u$ a $s_2 = u u_2$, řešením bude přímo slovo u_1, pokud platí $u_1 = u_2$. To znamená, že jsou-li slova p_1 a s_2 konjugovaná, má instance řešení.
Naopak, máme-li řešení $w = p_1 v$ pro (možná nulové) slovo v, je potom splněno $p_1 g(p_1 v) = h(p_1 v) s_2$, to znamená $p_1 p_1 v = p_1 v s_2$, tedy opět p_1, s_2 jsou konjugovaná a řešením je také slovo $u_1 = u_2$ při využití dříve zavedeného označení. V tomto případě tedy existuje řešení, právě když jsou p_1 a s_2 konjugovaná.

Tvrzení 4.1. *Mějme instanci $I = ((p_1, \epsilon), g, h, (\epsilon, s_2))$, kde*

$$\begin{aligned} g(a) &= a, \quad g(b) = b, \\ h(a) &= b, \quad h(b) = a. \end{aligned}$$

Pak její nejkratší řešení je dlouhé nejvýš $|p_1^2|$.

Důkaz. K existenci řešení musí mít slova p_1 a s_2 stejnou délku a $p_1 \neq \epsilon$. Pro slovo u označíme $\tilde{u} = h(u)$ a mějme řešení w, které je delší než uvedená mez. Potom existuje slovo x, $|x| > |p_1|$, takové, že $g(w) = x s_2 = w$, $h(w) = p_1 x = \tilde{w}$. Takže slovo w má tvar $w = \tilde{p}_1 \tilde{x} = x s_2$ a platí $\tilde{p}_1 < x$. Označme z_1 slovo splňující $\tilde{p}_1 z_1 = x$. Potom tedy platí

$$\tilde{p}_1 p_1 \tilde{z}_1 = \tilde{p}_1 z_1 s_2$$

$$p_1 \tilde{z}_1 = z_1 s_2$$

To znamená, že i slovo z_1 je řešením, přitom $|z_1| < |x|$. Pokud by i slovo z_1 bylo delší než $|p_1|$, lze znovu řešení zkrátit. Proto je nejkratší řešení délky nejvýš $|p_1^2|$. □

Tvrzení 4.2. *Mějme instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$, kde oba homomorfismy zachovávají délku. Potom lze v polynomiálním čase rozhodnout o existenci řešení.*

Důkaz. Pro identitu je nejkratší řešení dlouhé nejvýš $|p_1|$, pro druhý případ homomorfismů zachovávajících délku (až na symetrii) je dle Tvrzení 4.1 délka nejkratšího řešení nejvýš $|p_1^2|$. □

Zkusme hledat řešení pro instance se stálou sufixovou složitostí, jejichž homomorfismy však nezachovávají délku. Uvidíme, že délka jejich řešení je také omezena, a proto lze nalézt maximálního dostatečného kandidáta a následně i řešení v polynomiálním čase.

Příklad 10. Mějme instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$, kde

$$\begin{aligned} g(a) &= a^j \quad g(b) = b^k \\ h(a) &= b^m \quad h(b) = a^l, \end{aligned}$$

přitom NSD(j, m) =NSD(j, l) =NSD(k, m) =NSD$(k, l) = 1$. Nechť $p_1 = a^{\alpha_1} b^{\beta_1}$, $p_2 = \epsilon$, $s_1 = \epsilon$.
Hledejme nyní maximálního kandidáta na řešení w. Slovo p_1 určuje nutný začátek hledaného řešení. Víme tedy, že slovo w má prefix $b^{\alpha_1/l} a^{\lceil \beta_1/m \rceil}$. Srovnáním obrazů tohoto počátku řešení lze řešení postupně prodlužovat, nakonec dostaneme:

$$h(w)s_2 = a^{\alpha_1} b^{\beta_2} a^{\beta_2 \frac{j}{m}} b^{\beta_2 jk/lm} a^{\beta_2 j^2 k/lm^2} \ldots,$$

kde $\beta_2 = \beta_1 + \alpha_1 \frac{k}{l}$. Aby w mohlo být řešení, slovo s_2 musí mít pro nějaké $i \in \mathbb{N}_0$ jeden z tvarů:

$$a^{\beta_2 \frac{j}{m} (\frac{jk}{lm})^i - \gamma l} b^{\gamma k}, \text{ nebo} \quad b^{\beta_2 (\frac{jk}{lm})^i - \delta m} a^{\delta j},$$

přitom $\gamma, \delta \in \mathbb{N}_0$ a

$$\gamma l < \beta_2 \tfrac{j}{m} (\tfrac{jk}{lm})^i, \quad \gamma k < \beta_2 (\tfrac{jk}{lm})^{i+1} \text{ a}$$

$$\delta m < \beta_2 (\tfrac{jk}{lm})^i, \qquad \delta j < \beta_2 (\tfrac{jk}{lm})^i \tfrac{j}{m}.$$

Vidíme, že i je nejvýš takové, aby $\beta_2 (\frac{jk}{lm})^i$ bylo ještě celé číslo. Protože společný dělitel lm a jk je 1, je tedy číslo i jk-valuací čísla β_2 (tím myslíme, že i je nejvyšší takové, že $(jk)^i$ dělí β_2).

Tvrzení 4.3. *Mějme instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$, kde*

$$\begin{aligned} g(a) &= a^j, \quad g(b) = b^k, \\ h(a) &= b^m, \quad h(b) = a^l, \end{aligned}$$

přitom NSD(j, m) =NSD(j, l) =NSD(k, m) =NSD$(k, l) = 1$, max$(j, k, l, m) > 1$ a nechť je dvojice homomorfismů (g, h) balancovaná. Potom lze řešení nalézt v polynomiálním čase.

Důkaz. Všimněme si, že pokud $p_1 = p_2$ nebo $s_1 = s_2$, instance nemůže mít nenulové řešení. Kdyby totiž $p_1 = p_2$, nebudou slova $p_1 g(c)$ a $p_2 h(c)$ porovnatelná pro žádné písmeno c, obdobně i pro $s_1 = s_2$. Můžeme bez újmy na obecnosti předpokládat, že $p_1 \neq \epsilon$, $p_2 = \epsilon$.
Navíc pokud budeme řešení konstruovat postupně jako při hledání maximálního kandidáta, nikdy nedojdeme ke speciální situaci. Předpokládejme pro spor, že speciální situace nastane, tj. existuje w takové, že $p_1 g(w) = h(w)$. Potom tedy

$$|p_1|_a + |p_1|_b + j|w|_a + k|w|_b = l|w|_a + m|w|_b,$$

takže $|p_1|_a + j|w|_a = l|w|_a$ a $|p_1|_b + k|w|_b = m|w|_b$, což je spor s tím, že je pár (g, h) balancovaný.

Proto při hledání maximálního dostatečného kandidáta nenastane žádné větvení, pokračování lze vždy určit jednoznačně podle převisu obrazů nebo přímo slov předřešení. Navíc, jak ukazoval předešlý příklad, nelze předřešení sestrojovat nekonečně dlouho, neboť po určitém počtu kroků dojdeme ke sporu.
Mějme bez újmy na obecnosti $p_1 \bowtie a^s$ pro nějaké s. Potom musí maximální kandidát na řešení začínat slovem $b^{s/l}$. Pokud ovšem l nedělí s, bude maximální kandidát již slovo $b^{\lceil s/l \rceil}$. V příštím kroku prodloužíme maximálního kandidáta o $a^{sk/lm}$. K předřešení tedy vždy přidáváme střídavě bloky písmen a a b a délky těchto přidávaných bloků budou tedy

$$\frac{s}{l}\left(\frac{kj}{lm}\right)^{i-1}\frac{k}{m}, \text{ resp. } \frac{s}{l}\left(\frac{kj}{lm}\right)^{i},$$

kde $i > 0$ značí počet přidaných dvojic bloků. Z předpokladů nesoudělnosti tedy plyne, že pokud $(lm)^i$ nedělí s, nelze již přidat celý blok. Hodnota i je nejvýš $\sqrt{s}$, neboť nemůže být kvůli balancovanosti $l = m = 1$. Proto přidáme nejvýš $\sqrt{s}$ bloků, než dostaneme maximálního dostatečného kandidáta na řešení. □

4.2 Nebalancovaný pár homomorfismů

Pokud máme nebalancovaný (a neperiodický) pár homomorfismů, dává nám následující Tvrzení 4.4 omezení na délku případného řešení a můžeme jej zkusit hledat postupně.

Tvrzení 4.4. *Mějme instanci $I = ((p, q), g, h, (s, t))$, kde neperiodický pár (g, h) není balancovaný, a w jeho nejkratší řešení. Pak $|w| < M$ pro nějaké M polynomiálně velké vzhledem k $|I|$.*

Důkaz. Nechť tedy pro nějaké $\rho(.) \in \{|.|, |.|_a, |.|_b\}$ jsou zároveň splněné nerovnosti

$$\rho(g(a)) < \rho(h(a)) \tag{4.1}$$

$$\rho(g(b)) \leq \rho(h(b)) \tag{4.2}$$

Protože w je řešení, musí být $\rho(pg(w)s) = \rho(qh(w)t)$, a tedy:

$$\rho(p) + \rho(g(w)) + \rho(s) = \rho(q) + \rho(h(w)) + \rho(t)$$

$$\rho(ps) - \rho(qt) = \rho(h(w)) - \rho(g(w)). \tag{4.3}$$

Snadno ověříme, že platí $\rho(g(u)) = |u|_a\rho(g(a)) + |u|_b\rho(g(b))$ pro jakékoliv slovo u, obdobný vztah platí samozřejmě i pro $h(u)$. Použijeme tyto rovnosti pro $u = w$, poté dosazením do rovnice (4.3) obdržíme:

$$\rho(ps) - \rho(qt) = |w|_a\rho(h(a)) + |w|_b\rho(h(b)) - |w|_a\rho(g(a)) - |w|_b\rho(g(b))$$

$$\rho(ps) - \rho(qt) = |w|_a(\rho(h(a)) - \rho(g(a))) + |w|_b(\rho(h(b)) - \rho(g(b)))$$

$$\rho(ps) - \rho(qt) \geq |w|_a(\rho(h(a)) - \rho(g(a))),$$

neboť je jistě $|w|_b \geq 0$ a ze vztahu (4.2) máme $\rho(h(b)) - \rho(g(b)) \geq 0$.

Navíc díky rovnici (4.1) je spněno $\rho(h(a)) - \rho(g(a)) \neq 0$, můžeme tedy tímto výrazem poslední rovnost vydělit:

$$\frac{\rho(ps) - \rho(qt)}{\rho(h(a)) - \rho(g(a))} \geq |w|_a.$$

Tím tedy dostáváme omezení na počet písmen a v řešení nebalancované instance, označme tuto mez m_a.
Pro omezení na počet písmen b stačí uvažovat délky příslušných slov.
Pokud je $|g(b)| \neq |h(b)|$, dostaneme obdobně jako z rovnice (4.3):

$$|ps| - |qt| = |w|_a(|h(a)| - |g(a)|) + |w|_b(|h(b)| - |g(b)|),$$

takže omezení m_b pro počet písmen b v minimálním řešení bude mít hodnotu

$$m_b \leq \frac{|ps| - |qt|}{|h(b)| - |g(b)|} + m_a \frac{||g(a)| - |h(a)||}{|h(b)| - |g(b)|}.$$

Je-li naopak $|g(b)| = |h(b)|$, musíme postupovat jinak. Zkusíme spočítat, jaký je maximální počet po sobě jdoucích písmen b v minimálním řešení. Mějme tedy řešení $w = w_1 b^i w_2$ pro nějaké $i \geq 0$. Potom kdyby bylo $i > 0$ a převis slov $p_1 g(w_1 b^{i-1})$ a $p_2 h(w_1 b^{i-1})$ byl kratší než $|g(b^{i-1})| = |h(b^{i-1})|$, může být řešením i slovo $w_1 b^{i-1} w_2$. Proto vezměme i nejmenší takové, že

$$||p_1 g(w_1 b^i)| - |p_2 h(w_1 b^i)|| > |g(b^i)| = i|g(b)|.$$

Pak tedy

$$i < \frac{||p_1 g(w_1 b^i)| - |p_2 h(w_1 b^i)||}{|g(b)|}$$

$$i < \frac{||p_1| - |p_2|| + |w_1|_a(||g(a)| - |h(a)||)}{|g(b)|}$$

$$i < \frac{||p_1| - |p_2|| + m_a(||g(a)| - |h(a)||)}{|g(b)|} = m$$

a počet písmen b v minimálním řešení je omezen hodnotou $m_b = (m_a + 1)m$. □

4.3 Periodické GPCP

Do této chvíle jsme se zabývali pouze neperiodickými a nenulujícími instancemi, tj. ani jeden z homomorfismů nebyl periodický a žádný obraz žádného písmene nebyl nulový. Nulující homomorfismy považujeme také za periodické, za periodu můžeme vzít kořen delšího z obrazů obou písmen, resp. prázdné slovo, zobrazuje-li homomorfismus obě písmena na prázdné slovo.
Pro periodické instance platí $g(ab) = g(ba)$ a $h(ab) = h(ba)$ a nemůžeme aplikovat předchozí úvahy, které rozhodovaly podle přesahů za slovy z_g, z_h. Na druhou stranu periodicita nám dává informaci o obrazu případného řešení. V případě obou periodických homomorfismů jistě musí existovat nějaké pravidlo pro délky všech obrazů, aby řešení mohlo existovat, nejkratší pak lze omezit.
Situaci budeme řešit jinak, pokud je periodický pouze jeden z homomorfismů a jinak, jsou-li periodické oba. Pro každý z případů uvedeme jedno tvrzení, které ukazuje polynomialitu při hledání slova, které je obdobou maximálního dostatečného kandidáta.

Tvrzení 4.5. *Mějme instanci $I = ((p,q),g,h,(s,t))$, kde homomorfismy g, h jsou periodické. Pak lze GPCP(I) rozhodnout v polynomiálním čase.*

Důkaz. Mějme tedy homomorfismy dané rovnostmi:

$$\begin{aligned} g(a) &= u^k, \quad g(b) = u^l, \\ h(a) &= v^m, \quad h(b) = v^n \end{aligned}$$

pro jejich kořeny u a v. Ve slově w, které by mělo být řešením, nezáleží přímo na pořadí písmen a a b, pouze na jejich počtu. Předpokládejme, že w je nejkratší řešení a označme $\alpha = |w|_a$, $\beta = |w|_b$. Díky symetrii můžeme předpokládat, že druhé počáteční slovo je kratší, tedy $q \leq p$.

1. Jestliže platí $pu^\infty = qv^\infty$, podle Tvrzení 1.3 a Lemmatu 1.1 jsou slova u a v konjugovaná, tedy $u = xy$ a $v = yx$ pro nějaká různá slova x, y. Potom, je-li $h(w) \geq |q^{-1}pu|$, nutně $q^{-1}p = (yx)^i y$ pro nějaké $i > 0$ a pro nějaké $j \geq 0$

$$\begin{aligned} st^{-1} &= x(yx)^j = (xy)^j x, \quad \text{v případě, že } t \leq_s s, \\ ts^{-1} &= y(xy)^j = (yx)^j y, \quad \text{jinak.} \end{aligned}$$

Potom tedy má platit:

$$q^{-1}pg(a^\alpha b^\beta)st^{-1} = h(a^\alpha b^\beta), \quad \text{resp.} \quad q^{-1}pg(a^\alpha b^\beta) = h(a^\alpha b^\beta)ts^{-1}.$$

Porovnání délek těchto slov tedy dostaneme v prvním případě:

$$i|xy| + |y| + \alpha|xy|k + \beta|xy|l + j|xy| + |x| = \alpha|yx|m + \beta|yx|n$$

$$i + 1 + \alpha k + \beta l + j = \alpha m + \beta n$$

a ve druhém případě bude

$$i + \alpha k + \beta l = \alpha m + \beta n + j.$$

Máme-li $k = m$ nebo $l = n$, vyřešíme rovnici pro druhou neznámou a řešením bude $w = b^\beta$, resp. a^α. Pokud jsou obě čísla $k - m$ i $l - n$ nenulová, vidíme, že rovnice $\alpha(m-k) + \beta(n-l) = j + i + 1$ bude mít řešení právě tehdy, když $\mathrm{GCD}(m-k, n-l)$ dělí $j+i+1$. Platnost této podmínky lze ověřit v polynomiálním čase, vypočítat konkrétní α a β lze například s využitím rozšířeného Euklidova algoritmu. Řešení w, pro které by $|h(w)| < |q^{-1}pu|$, můžeme vyzkoušet všechna.

2. Jestliže se daná slova nerovnají neomezeně dlouho, tj. $q^{-1}pu \neq vq^{-1}p$, existuje nějaké konečné slovo $u_1 = pu^\infty \wedge qv^\infty$. To má podle Lemmat 1.2, 1.4 omezenou délku $|u_1| < \max(|p|,|q|) + \max(|u^2|,|v^2|)$. Má-li být slovo w řešením, musí počty jeho písmen $\alpha = |w|_a$ a $\beta = |w|_b$ splňovat pro $s \leq t$ nerovnost

$$|h(w)| = \alpha|h(a)| + \beta|h(b)| \leq |u_1| - |q|,$$

pokud je delší druhé z koncových slov, tak

$$|g(w)| = \alpha|g(a)| + \beta|g(b)| \leq |u_1| - |p|.$$

Vzhledem k tomu, že koeficienty obou nerovnic jsou nezáporné, jistě

$$0 \leq \alpha, \beta \leq \max(|u_1| - |p|, |u_1| - |q|),$$

stačí tedy ověřit, zda pro některou z dvojic α, β splňující příslušnou nerovnost bude $a^\alpha b^\beta$ řešením. □

Tvrzení 4.6. *Mějme instanci $I = ((p, q), g, h, (s, t))$, kde homomorfismus h je periodický a g neperiodický. Pak lze GPCP(I) rozhodnout v polynomiálním čase.*

Důkaz. Vezměme u jako maximální slovo splňující $pg(u) \leq qv^\infty$.

1. Je-li $|u| < \infty$, vezmeme u_1 jeho nejdelší prefix splňující $pg(u_1)z_g < qv^\infty$. Takové slovo je maximálním kandidátem na řešení a stačí vyzkoušet, zda některý jeho prefix prodloužený o slovo $w \in F^g$, nedává řešení. Přitom je slovo $g(u_1)$ dlouhé nejvýš $|v|$, pokud $p \geq q$, a nejvýš $|q| + |v|$ v druhém případě, jinak by vznikl stejný převis dvakrát a slovo u by bylo nekonečné.

2. Jestliže $|u| = \infty$, musí při jeho vytváření pomocí převisů dojít nekonečněkrát za sebou ke stejnému převisu (tímto také rozlišíme situace 1. a 2.). To znamená, že $(pg(u_1))^{-1}pg(u) = (pg(u_2))^{-1}pg(u)$ pro dva různé prefixy $u_1 < u_2 < u$ dlouhé aspoň tak, aby $pg(u_1) > q$. Musí tedy být $u = u_1x^\infty$, kde $x = u_1^{-1}u_2$. Rozdělíme minimální řešení w instance I na dvě slova: $w = w_1w_2$ tak, že $|g(w_2)| \leq |z_g|$. Potom vidíme, že $w_1 < u_1x^k$ pro nějaké $k \geq 0$.
Pokud $|g(x)| \neq |h(x)|$, lze z rovnice $|pg(w_1w_2)s| = |qh(w_1w_2)t|$ získat omezení

$$k \leq \frac{|qt| - |ps| + |h(u_1w_2)| - |g(u_1w_2)|}{|g(x)| - |h(x)|}.$$

Je-li $|g(x)| = |h(x)|$, pak stačí volit k, aby platilo:

$$|p| \leq |qh(u_1x^{k-1}u')|$$

$$|q| \leq |pg(u_1x^{k-1}u')|,$$

kde $u' \leq x$. Převisy slov $pg(u_1x^{k-1}u')$ a $pg(u_1x^ku')$ jsou shodné, takže pro nejkratší řešení musí být již $w_1 \leq u_1x^k$. □

5. Detailní popis algoritmu

Díky poznatkům z předcházejících kapitol lze nyní algoritmus popsat podrobně. Poznali jsme, že průběh bude zcela odlišný v případě periodické a neperiodické instance. Navíc se práce prozatím věnovala především zobecněnému problému, poslední část této kapitoly vysvětluje, jak řešit PCP. Začněme nejdříve rozborem řešení neperiodické instance. Doporučujeme při čtení této kapitoly využít obrázky blokových schémat těchto algoritmů, které jsou v příloze.

5.1 Neperiodické GPCP

Algoritmus pro neperiodické GPCP vyadřuje také obrázek 6.2. Mějme na vstupu instanci $I = ((p_1, p_2), g, h, (s_1, s_2))$, kde g a h jsou neperiodické homomorfismy, p_1, p_2 porovnatelná a $\overline{s_1}$, $\overline{s_2}$ také porovnatelná.

Nejprve zkusíme nalézt počáteční blok této instance. Začneme dvojicí slov $(w_g, w_h) = (\epsilon, \epsilon)$. Převis slov $(p_1 g(w_g) z_g, p_2 h(w_h) z_h)$ určuje jednoznačné prodloužení jednoho ze slov w_g a w_h, pokud nenastane jedna z těchto situací:

1. po přidání písmene nastane spor mezi obrazy (tj. nebylo by porovnatelné $p_1 g(w_g) z_g$ a $p_2 h(w_h) z_h$),
2. po přidání písmene by nebylo porovnatelné w_g a w_h,
3. narazíme na kritický převis, nelze tedy rozhodnout o pokračování tímto způsobem,
4. dojdeme podruhé k totožnému převisu.

Dvojici (w_g, w_h) tedy prodlužujeme, dokud nedojde k některému ze zmíněných případů. Ve třetím případě jsme nalezli počáteční blok, ve zbylých tento blok neexistuje. Pokud instance nemá počáteční blok, nemůže být řešení rozložitelné na bloky a stačí najít maximálního dostatečného kandidáta na řešení I, který je porovnatelný s nalezenými slovy w_g, w_h. Z něj potom zjistíme případné řešení zkusmým přidáváním finálních slov ke všem jeho prefixům. Hledání maximálních dostatečných kandidátů je podrobně rozebráno níže, neboť může být využito na více místech celého algoritmu.

Pokud máme již nalezený počáteční blok, nic nám nebrání zkusit, jak by řešení mohlo pokračovat dál. Samozřejmě se nabízí i možnost vyhledat písmenné bloky a koncové bloky a poté přejít k následníkovi, je-li to možné. Jestliže však zkusíme prodloužit počáteční blok, aby mohl být ještě řešením, také jeden z písmenných bloků nalezneme, navíc s jistotou, že bude navazovat na počáteční blok (viz Příklad 11). Tímto postupem vyloučíme možnost, že se budeme zbytečně zabývat posloupností následníků, pokud na počáteční blok (w_g, w_h) nelze navázat písmenným blokem (x, y), aby slova $w_g x$, $w_h y$ byla porovnatelná.
Stejným postupem jako při hledání počátečního bloku tedy hledáme navazující písmenný blok. Pokud takový neexistuje, dohledáme z nalezených slov $w_g x$,

$w_h y$ maximálního dostatečného kandidáta a vyzkoušíme, zda některý jeho prefix prodloužený o finální slovo nedává řešení.

Příklad 11. Mějme instanci $I = ((\epsilon, \epsilon), g, h, (s_1, s_2))$, kde s_1 a s_2 jsou jakákoliv sufixově porovnatelná slova a homomorfismy jsou určeny hodnotami:

$$\begin{aligned} g(a) &= ab, \quad & g(b) &= babb, \\ h(a) &= bab, \quad & h(b) &= abbab. \end{aligned}$$

Snadno zjistíme, že počáteční blok je dvojice prázdných slov a písmenné bloky existují a mají tvar (aba, ba), (ba, aa). Mohli bychom tedy ještě nalézt koncové bloky, případně následníky redukovat do jednoho, a hledat řešení pro následnickou instanci.
Pokud však začneme hledáním maximálního dostatečného kandidáta, první kritický převis nastává hned při předřešení $(w_1, w_2) = (\epsilon, \epsilon)$, což znamená, že jsme nalezli počáteční blok. Zároveň máme speciální situaci. Přidáme-li tedy k w_1 písmeno a, vede nás převis k přidání písmene b k druhému slovu předřešení a ta tak budou neporovnatelná. Ve druhé větvi začínající písmenem b je stejný problém. Maximální kandidáti jsou tedy pouze slova a a b.

Po nalezení počátečního bloku by ale mohl vzniknout problém s rozhodnutím jak nalezený pár (w_g, w_h) prodloužit, to v případě $w_g = w_h$. Vyzkoušíme tedy obě možnosti - přidání písmene a k oběma slovům a přidání písmene b k oběma slovům. Buď takto můžeme navázat na předřešení oběma písmennými bloky, nebo se dostaneme do stejných situací, jako kdyby speciální situace nenastala, pouze si musíme pamatovat, že již nemůže nastat žádná další speciální situace při hledání maximálního dostatečného kandidáta.

Tento počátek řešení zajistí funkce PRIDAVEJ vyjádřená schématem na Obrázku 6.1. Nejprve je spuštěna pro $(w_g, w_h) = (\epsilon, \epsilon)$. Pokud je vráceno $z = c(g, h)$, máme počáteční blok (w_g, w_h). Pokud jsou slova počátečního bloku různá, spustíme tuto funkci s počátečním blokem na vstupu. V opačném případě ji spustíme dvakrát - jednou pro $(w_g a, w_h a)$ a podruhé pro $(w_g b, w_h b)$. Takto popsaným způsobem řídí funkci PRIDAVEJ funkce nazvaná TESTBLOKU, jejímž výstupem je počáteční blok (w_g, w_h), případný první navazující blok (x, y) a vzniklý převis α.

Máme-li navazující písmenný blok (x, y), zkusíme najít druhý písmenný blok, tj. slova (e, f), že $\mathrm{pref}_1(e) \neq \mathrm{pref}_1(x)$, $\mathrm{pref}_1(f) \neq \mathrm{pref}_1(y)$, splňující

$$z_h g(e) z_g = z_g h(f) z_h.$$

Toto lze provést obdobně přidáváním písmen podle převisů jako v předešlých situacích. (Možná jsme však tento blok již odhalili, pokud $w_g = w_h$ a oběma písmennými bloky šlo navázat na počáteční blok, aby slova byla porovnatelná.) Jestliže tento druhý písmenný blok neexistuje, můžeme hledat maximálního dostatečného kandidáta prodlužováním dvojice $(w_g x, w_h y)$. Dohledávání maximálních dostatečných kandidátů je dopodrobna popsané níže, neboť jej v algoritmu využíváme na více místech.

Předpokládejme tedy, že má instance I oba písmenné bloky a také počáteční blok. Stejným způsobem také zjistíme, zda tyto bloky existují pro reversní instanci $\overline{I}$. Jestliže některý z nich neexistuje, najdeme příslušné maximální dostatečné kandidáty této reversní instance a zkusíme dohledat řešení. Řešení původní instance existuje, právě když existuje řešení reversní instance, tato slova jsou totiž pouze zrcadlově obrácená.

Příklad 12. Mějme instanci $I = ((ab, a), g, h, (\epsilon, aba))$ pro

$$\begin{aligned} g(a) &= aba, \quad & g(b) &= bb, \\ h(a) &= b, \quad & h(b) &= aba. \end{aligned}$$

Již uvedeným způsobem zjistíme počáteční a písmenné bloky této instance, totiž $(p'_1, p'_2) = (\epsilon, a)$ a

$$\begin{aligned} g_1(a) &= a, \quad & g_1(b) &= b, \\ h_1(a) &= b, \quad & h_1(b) &= aa. \end{aligned}$$

Podívejme se na tuto instanci tedy odzadu. Pro reversní instanci

$$I' = ((\epsilon, aba), \overline{g} = g, \overline{h} = h, (ba, a))$$

zkusíme také nalézt všechny tři bloky. Počáteční blok (a, ϵ) je naším předřešením (w_g, w_h). První slovo tohoto bloku je delší, proto přidáme i k w_h písmeno a. Nyní podle převisu jejich obrazů, tj. podle převisu slov $g(a)$, $h(a)$ poznáme, že máme přidat písmeno b k w_g. Převis obrazů by nás nyní zavedl k přidání písmena a ke druhému slovu předřešení, čímž by se slova w_g, w_h stala neporovnatelnými. Proto je slovo ab maximálním dostatečným kandidátem na řešení instance I'. Reversní instance sice má následníka i počáteční blok, jenže nelze žádným blokem na tento počáteční navázat. Množina finálních slov obsahuje pouze prázdné slovo, řešení však existuje, je to slovo a, které je tedy zároveň i řešením původní instance, neboť $\overline{a} = a$.

Zbývá popsat postup pro instanci I, která má oba písmenné bloky i počáteční blok a to samé platí i pro její reversní instanci. V tomto případě nalezneme všechny koncové bloky instance I. Jestliže žádný neexistuje, je maximálním dostatečným kandidátem delší ze slov počátečního bloku, neboť nemůže existovat řešení rozložitelné na bloky. Pokud nějaké koncové bloky existují, ověříme, zda je instance balancovaná a nemá stálou sufixovou složitost. V opačném případě známe omezení na délku maximálního dostatečného kandidáta. Jsou-li tyto podmínky splněny, můžeme ještě zkusit najít nedlouhé řešení. V případě neexistence nedlouhého řešení nezbývá než zredukovat následníky do jednoho (pokud to lze) a zopakovat celý postup pro něj.

Příklad 13. Ukážeme si nyní instanci, která má nedlouhé řešení. Mějme

$$\begin{aligned} g(a) &= baba, \quad & g(b) &= bb, \\ h(a) &= b, \quad & h(b) &= ababb \end{aligned}$$

a instanci $I = ((a, a), g, h, (\epsilon, \epsilon))$.
Úvod algoritmu nalezne počáteční blok $(p'_1, p'_2) = (\epsilon, a)$ a oba písmenné bloky:

$(e_a, f_a) = (ab, ba)$ a $(e_b, f_b) = (b, aa)$. Reversní instance také má všechny tři bloky, vyhledáme tedy koncové bloky naší instance. Koncový blok existuje jediný: $(u, v) = (b, a)$. Nedlouhá řešení jsou dlouhá nejvýš $\max\{|p'_1 e_a e_b u|, |p'_2 f_a f_b v|\} = 6$. Zkusíme tedy dosud nalezené předřešení prodloužit na maximálního dostatečného kandidáta délky nejvýš šest. Snadno dojdeme k řešení ab a nemusíme vůbec přecházet k následnické instanci.

Pro úplnost je ještě uveden popis funkce, která vyhledává maximální dostatečné kandidáty zadané instance, a její schéma (6.3). Na jejím vstupu je kromě instance I také dvojice porovnatelných slov w_g, w_h, která chceme na maximální dostatečné kandidáty prodloužit (mohou to být i prázdná slova), číslo n označující počet speciálních situací ve dvojici slov na vstupu a mez omezující délku maximálního dostatečného kandidáta. Tato mez bude konečná pouze v případě, že známe nějaké omezení na délku řešení, tedy u nebalancovaných párů a instancí se stálou sufixovou složitostí a také u nedlouhých řešení.

Máme-li tedy dvojici porovnatelných slov (w_g, w_h), přidáváme k ní další písmena podle převisu, dokud nenastane jedna z těchto situací:
1) Nelze přidat další písmeno, protože by po prodloužení nebyla splněna jedna z podmínek $p_1 g(w_g) z_g \bowtie p_2 h(w_h) z_h$ a $w_g \bowtie w_h$. V tomto případě je tedy delší ze slov w_g, w_h maximálním dostatečným kandidátem.
2) Převis je kritický, tj. $p_1 g(w_g) z_g = p_2 h(w_h) z_h$.
2a) Nechť je navíc $w_g = w_h$ (tzv. speciální situace). Nastala-li tato situace poprvé, spustíme algoritmus znovu dvakrát - pro dvojici $(w_g a, w_h a)$ a dvojici $(w_g b, w_h b)$. Jestliže nastala speciální situace již podruhé, máme maximálního dostatečného kandidáta.
2b) Nenastává speciální situace, bez újmy na obecnosti je $w_g < w_h$. Potom w_g prodloužíme o první písmeno z převisu w_g, w_h, tedy první písmeno slova $w_g^{-1} w_h$, a spustíme algoritmus znovu pro tuto dvojici.
3) Narazíme podruhé na stejný převis. Poté prodloužíme slova w_g, w_h tak, aby platilo $p_1 g(w_g) > p_2$ a $p_2 h(w_h) > p_1$, a najdeme opakující se blok (x, y), tedy takový, že převis slov $p_1 g(w_g), p_2 h(w_h)$ je shodný s převisem $p_1 g(w_g x), p_2 h(w_h y)$. Jestliže pro nějakou neprázdnou dvojici slov $(x', y') < (x, y)$ platí, že

$$p_1 g(w_g x') z_g = p_2 h(w_h y') z_h,$$

prodloužíme w_g o x' a w_h o y' a nalezneme opět opakující se blok (x, y) k těmto slovům.
Jestliže lze tento blok přidat nekonečněkrát aniž by došlo ke sporu v porovnatelnosti slov $p_1 g(w_g x^\infty)$, $p_2 h(w_h y^\infty)$ či vzorů $w_g x^\infty$, $w_h y^\infty$, je v případě $|x| = |y|$ maximálním dostatečným kandidátem již slovo $\max(w_g x, w_h y)$. Pokud se však délky x a y liší, budou se přidáváním bloku (x, y) délky slov předřešení k sobě blížit a poté oddalovat. Maximální dostatečný kandidát je tedy nalezen ve chvíli, kdy po přidání bloku (x, y) k (w_g, w_h) nastane $||w_g x| - |w_h y|| > ||w_g| - |w_h||$ a také $||w_g x| - |w_h y|| > |z_g|, |z_h|$. Tento poslední případ ukazuje Příklad 14.
V opačném případě můžeme spustit rekurzivně algoritmus hledání maximálních dostatečných kandidátů na $w_g x$, $w_h y$, neboť po konečném počtu kroků dojde k jeho ukončení popsaném v bodě 1).

Příklad 14. Mějme instanci $((\epsilon, \epsilon), g, h, (\epsilon, \epsilon))$ danou

$$\begin{aligned} g(a) &= a, & g(b) &= bb, \\ h(a) &= abbbbbb = ab^6, & h(b) &= b. \end{aligned}$$

Oba homomorfismy jsou markované, máme tedy $z_g = z_h = \epsilon$. Začněme konstruovat řešení písmenem a, prvním předřešením bude tedy dvojice $(w_1, w_2) = (a, a)$. Dle vznikajících převisů prodlužujeme slovo w_1, dokud nedojdeme ke kritickému převisu. V tomto bodě je předřešením $(abbb, a)$.
Druhé slovo z předřešení tedy musí pokračovat písmenem b. Další konstrukcí se dostaneme opět ke krtickému převisu, tentokrát má předřešení tvar $(abbbb, abb)$. Od prvního kritického převisu jsme tedy přidali dvojici (b, bb), převis předřešení nás opět vede k pokračování písmenem b, tentokrát u druhého slova. Ovšem „blok“, který bychom přidali, bude stejný: (b, bb).
Tento blok má však obě slova různě dlouhá, což způsobí, že se délky obou slov předřešení k sobě blíží. Po třetím přidání tohoto bloku budou délky shodné, dosáhneme tedy řešení ab^6.

5.2 Periodické GPCP

Předpokládejme, že na vstupu máme instanci I s alespoň jedním homomorfismem periodickým. Přitom algoritmus bude probíhat odlišně, pokud je periodický pouze jeden z homomorfismů nebo oba. Pro instance s jediným periodickým homomorfismem je algoritmus zobrazen na schématu 6.4.

Je-li periodický pouze jeden z homomorfismů, předpokládejme, že je to h a že oba jeho obrazy jsou mocninou slova u. Začneme opět s prázdným slovem w_g, které bude představovat zatím nalezený prefix jediného možného kandidáta na řešení, a prázdným slovem v_h, které uchovává vždy nejmenší mocninu slova u aby platilo $p_1 g(w_g) z_g < p_2 v_h$. Nejprve tedy přidáváme k v_h slovo u, aby $p_1 z_g < p_2 v_h$. Poté můžeme rozhodnout o prvním písmenu případného řešení podle převisu slov p_1 a $p_2 v_h$, toto písmeno přidáme ke slovu w_g. Tento postup opakujeme, dokud se nedostaneme k rozporu, tj. nelze přidat žádné písmeno ke slovu w_g, aby byla slova $p_1 g(w_g) z_g$ a $p_2 v_h$ porovnatelná, nebo dokud není slovo w_g dlouhé aspoň tak, aby $p_1 g(w_g) > p_2$ a $p_2 h(w_g) > p_1$.

V prvním případě již řešení může být složené jedině z prefixu slova w_g prodlouženého o slovo v, pro které $g(v) < |z_g|$, což snadno vyzkoušíme.

Ve druhé situaci prodlužujeme slova w_g a v_h nadále uvedeným způsobem, dokud nedostaneme dvakrát týž převis slov $p_1 g(w_g)$ a $p_2 v_h$. Pokud při tomto prodlužování nebude možné přidat žádné písmeno, aby byla porovnatelná slova $p_1 g(w_g) z_g$ a $p_2 v_h$, opět jediná možnost, jak by mohlo vypadat řešení zadané instance, je prefix slova w_g spolu s nějakým finálním slovem.
Jestliže nemůžeme prodloužit w_g a v_h uvedeným způsobem, víme podle uvedeného Tvrzení 4.6, že řešení už opět vznikne jedině z prefixu slova w_g a slova u, které je finálním slovem homomorfismu g.

Příklad 15. Mějme například instanci PCP $I = (g, h)$, kde

$$g(a) = abab, \quad g(b) = ababab,$$
$$h(a) = ababa, \quad h(b) = babab.$$

Homomorfismus g je periodický s kořenem $u = ab$. Dvojici (v_g, w_h) tedy nejprve určíme jako (u, ϵ). Převis obrazů těchto slov určuje jejich prodloužení na (u, a). Musíme tedy ke slovu $v_h = u$ přidat ještě slovo u^2, aby bylo slovo $g(v_g)$ delší než $h(w_h)$. Pak lze prodloužit w_h na ab. Je nutné přidat u^3 k v_h, čímž vznikne stejný převis jako u dvojice (u, ϵ). Poté musíme nalézt příslušné k z Tvrzení 3.3 a tím i maximálního dostatečného kandidáta.

Mějme nyní periodické oba homomorfismy uvedené instance, označme u kořen homomorfismu g a v kořen h. Je-li nějaké slovo w řešením této instance, je řešením i slovo, které je libovolnou permutací písmen slova w, proto stačí spočíst, kolik písmen a a kolik písmen b musí případné řešení obsahovat. Hledání řešení se zde opět rozděluje na dvě větve, podle toho, zda $p_1 u^\infty = p_2 v^\infty$. Pokud uvedená rovnost platí, lze konkrétní počet písmen a a b v řešení nebo alespoň existenci řešení zjistit z rovnic v Tvrzení 4.5. Pokud existuje $k < \infty$, že neplatí $p_1 u^k = p_2 v^k$, víme podle Tvrzení 4.5, že α počet písmen a i β počet písmen b v nejkratším řešení instance I je nejvýš $\max(m - |p|, m - |q|)$, kde $m = \max(|p|, |q|) + \max(|u^2|, |v^2|)$. Stačí tedy vyzkoušet pro všechny dvojice (α, β) splňujících uvedenou mez, zda je řešením slovo $a^\alpha b^\beta$.

5.3 Řešení PCP pomocí GPCP

Každá instance problému PCP je zároveň instancí GPCP, proto se zdá, že by pro řešení PCP měl fungovat již uvedený postup. Problém ovšem je, že instance GPCP může mít jako řešení i prázdné slovo, které však za řešení u instancí PCP nepovažujeme. Navíc často v popisovaném algoritmu konstruujeme řešení, dokud si nejsme jisti, že nejkratší řešení již nalezneme ze sestrojeného slova. Jenže pokud by toto nejkratší řešení bylo právě prázdné slovo, nevíme jistě, zda PCP řešení bude mít, či ne. Řešení pro periodickou instanci hledáme zcela analogicky jako pro instanci GPCP. Algoritmus pro hledání řešení neperiodického PCP rozdělíme na dvě větve podle markovanosti instance na vstupu.

Pokud není alespoň jeden z homomorfismů dané instance markovaný a navíc $c(g, h) > \epsilon$, nalezneme následníka této instance, což je již instance GPCP, a dále hledáme řešení tak, jak bylo popsáno výše. Nalezené řešení původní instance PCP se skládá alespoň z jeho počátečního bloku, který je nenulový. Jestliže následník neexistuje a existuje počáteční blok, může být řešením buď prefix delšího ze slov počátečního bloku, nebo počáteční blok s jedním z koncových bloků. Neexistuje-li počáteční blok, buď při jeho sestrojování došlo ke sporu a řešení může být jedině nějaký prefix delšího ze sestrojených slov počátečního bloku prodloužený o finální slovo, nebo došlo podruhé ke stejnému převisu. Potom ale již také není nutné nalezená slova prodlužovat, jak bylo popsáno při hledání maximálních dostatečných kandidátů.

Jsou-li oba homomorfismy markované nebo $z_g = z_h$, ověříme nejdřív, zda se nejedná o nebalancovanou instanci nebo instanci se stálou sufixovou složitostí. U těchto známe omezení na délku řešení, můžeme tedy sestrojit maximálního dostatečného kandidáta a dohledat příslušné řešení, jak bylo popsáno výše. Není-li instance ani jednoho z těchto typů, ověříme, zda neexistuje jednopísmenné řešení. Pokud ne, sestrojíme následníka této instance a opět ověříme, zda nemá řešení délky 1. Tento postup opakujeme, dokud nezískáme jednopísmenné řešení, potom totiž víme, že i původní instace má řešení, nebo neexistuje následník, to znamená, že řešení zadané instance neexistuje. Vzhledem k tomu, že sufixová složitost se snižuje, k jedné z těchto dvou situací dojde v uvedeném případě po polynomiálně mnoha krocích.

6. Dokumentace k aplikaci

Přiložená webová aplikace řeší binární Postův korespondenční problém v polynomiálním čase. Algoritmus je podrobně popsán v páté kapitole diplomové práce. Spuštění aplikace popisuje soubor info.txt, který je k práci přiložen. Aplikace zároveň poběží na webových stránkách http://kurip7am.asp2.cz/formular.aspx, kde zůstane nezměněna do dne obhajoby práce.

6.1 Struktura programu

Jedná se o webovou aplikaci naprogramovanou ve Visual Studio 2010 Professional. Obsahuje třídu prvek a samotný formulář.
Ve třídě prvek najdeme základní funkce a vlastnosti, které se přímo týkají dané instance GPCP. Formulář pak kromě popisu chování jednotlivých tlačítek obsahuje celý algoritmus tak, jak byl popsán.

6.2 Reprezentace vstupních dat

Zadání se vpisuje do levé horní části stránky (Obrázek 6.5). Políčko GPCP nechte zaškrtnuté, chcete-li řešit zobecněný Postův korespondenční problém. Pro řešení PCP jej odklikněte, aby bylo prázdné. Instanci, kterou chcete řešit, vepište do polí v levé horní části: do pole za „p1=“ napište první počáteční slovo, za „p2=“ druhé počáteční slovo, za „g(a)=“ obraz písmene „a“ při homomorfismu g atd. Je-li některé slovo prázdné, příslušné pole ponechte prázdné. Povolené znaky jsou jen 'a' a 'b'.

Tlačítko „spočti“ odešle vstupní data programu a zahájí výpočet. Pomocí tlačítek „krok“ a „vysledek“ se zobrazí výsledek buď po jednotlivých krocích, nebo rovnou konečný výstup. Dalším tlačítkem je „vymaž výstup“, které vymaže výstupní texty (pod oblastí se zadáním a zcela vpravo) a aplikace povolí přepsání a načtení nových dat. Poslední tlačítko „vymaž vstup“ vynuluje slova ze zadání vstupní instance, což může být užitečné, pokud podruhé zadáváme instanci, která je od první odlišná.

6.3 Reprezentace a interpretace výstupních dat

V levé dolní části na Obrázku 6.5 se vypisují výsledky jednotlivých kroků vždy po stlačení „krok“, resp. celkový výsledek po stisku „vysledek“. Typický výsledek po stisknutí tlačítka „krok“ může být například: „Instance nemá počáteční blok“, tlačítko výsledek pouze přidá k textu řádek oznamující, zda řešení instance existuje, nebo ne, případně i jak nějaké nalezené řešení vypadá.

Vpravo od polí pro zadání (viz. Obrázek 6.6) můžeme sledovat postup algoritmu po krocích - červená šipka vždy ukazuje, v jaké části algoritmu se výpočet právě nachází. Přitom některé bloky algoritmu nejsou v obrázku zcela podrobně,

například funkce PERIODICKE. U tohoto kroku tedy algoritmus nezobrazuje podrobné kroky při řešení periodické instance, rovnou vypíše, jak řešení vypadá.

Zcela vpravo se potom vypisuje postupně posloupnost následníků - na prvním řádku je zadaný prvek, na druhém jeho následník, na třetím následník tohoto následníka atd. Tato posloupnost se vypisuje jak při jednotlivých krocích, tak i při pouhém vypsání výsledků.

6.4 Program

Každá entita třídy prvek má přiřazené hodnoty jeho homomorfismů na obou písmenech, počáteční a koncová slova, množinu následníků, kritický převis a svoji reversní instanci. Mezi její funkce, které jsou podrobně popsány v diplomové práci, patří TestBloku, Pridej, Pridavej, Kandidat, Reseni, Periodicke, nekonecne, DruhyBlok, endBloky. Také zde najdeme rozhodovací algoritmy - balancovane, markovane, stalaSS, periodicke - odpovídají „true“, jsou-li příslušné vlastnosti (balancovanost, markovanost, stálá sufixová složitost, resp. periodicita) splněné. Poslední část tvoří pomocné algoritmy např. pro zjištění konjugovanosti dvou slov, rozšířený Euklidův algoritmus, výpočet hodnoty daného homomorfismu na konkrétním slově či výpočet kritického převisu.

Formulář uchovává posloupnost následníků v seznamu prvků „naslednici“ a jednotlivé kroky v seznamu čísel „kroky“. Při stisknutí tlačítka „spočti“ se provede celý výpočet algoritmu pro zadanou instanci a již nelze zadání měnit. Jednotlivé kroky algoritmu se při tom uloží pod svými číselnými kódy do seznamu „kroky“, případní následníci do seznamu následníků. Tlačítka „krok“ a „výsledky“ pouze zajišťují zobrazování spočtených výsledků.

Závěr

Cílem této práce bylo podrobně popsat algoritmus, který by řešil binární Postův korespondenční problém v polynomiálním čase. Stavět jsem začala na teorii známé z několika matematických článků, která je popsána také v počátečních kapitolách. Za klíčový pojem z této části bych označila především následníky homomorfismů a také tvrzení, která nám umožňují následníky redukovat do jednoho reprezentanta. Poznali jsme, jak se nahrazováním instancí jejich následníkem řešení zjednodušuje.

Jenže jak najít řešení poslední instance z posloupnosti následníků a co dělat, pokud následník vůbec neexistuje? Na to odpovídá Kapitola 3. Ta zavádí další důležité pojmy a pomocí nových tvrzení napovídá, jak v těchto případech rozhodovat. Navíc přesně popisuje, jak hledat řešení pomocí převisů, jehož myšlenka se často objevuje u člověka, který se poprvé snaží najít řešení nějaké konkrétní instance PCP. Přesný popis algoritmu však přenechává Kapitole 5, jeho implementaci potom webové aplikaci na adrese: http://kurip7am.asp2.cz/formular.aspx. V práci bylo popsáno řešení PCP pomocí zobecněného PCP, jehož řešení bylo také detailně rozebráno. Řešení GPCP je tedy vedlejším produktem této práce, webová aplikace také umožňuje volbu, zda chce uživatel řešit PCP nebo GPCP.

Seznam použité literatury

[1] Crochemore, Maxime, Hancart, Christophe, Lecroq, Thierry, *Algorithms on strings*, New York: Cambridge University Press, 2007, ISBN-13: 9780521848992

[2] Ehrenfeucht, Andrzej, Karhumäki, Juhani, Rozenberg, Grzegorz, *The (generalized) Post Correspondence Problem with lists consisting of two words is decidable*, Theoret. Comput. Sci., 21:119–144, 1982

[3] Halava, Vesa, Harju, Tero, Hirvensalo, Mika, *Binary (generalized) Post Correspondence Problem*, Theoret. Comput. Sci., 276:183–204, 2002

[4] Halava, Vesa, Holub, Štěpán, *Reduction Tree of the Binary Generalized Post correspondence Problem*, International Journal of Foundations of Computer Science, World Scientific Publishing Company, 2010

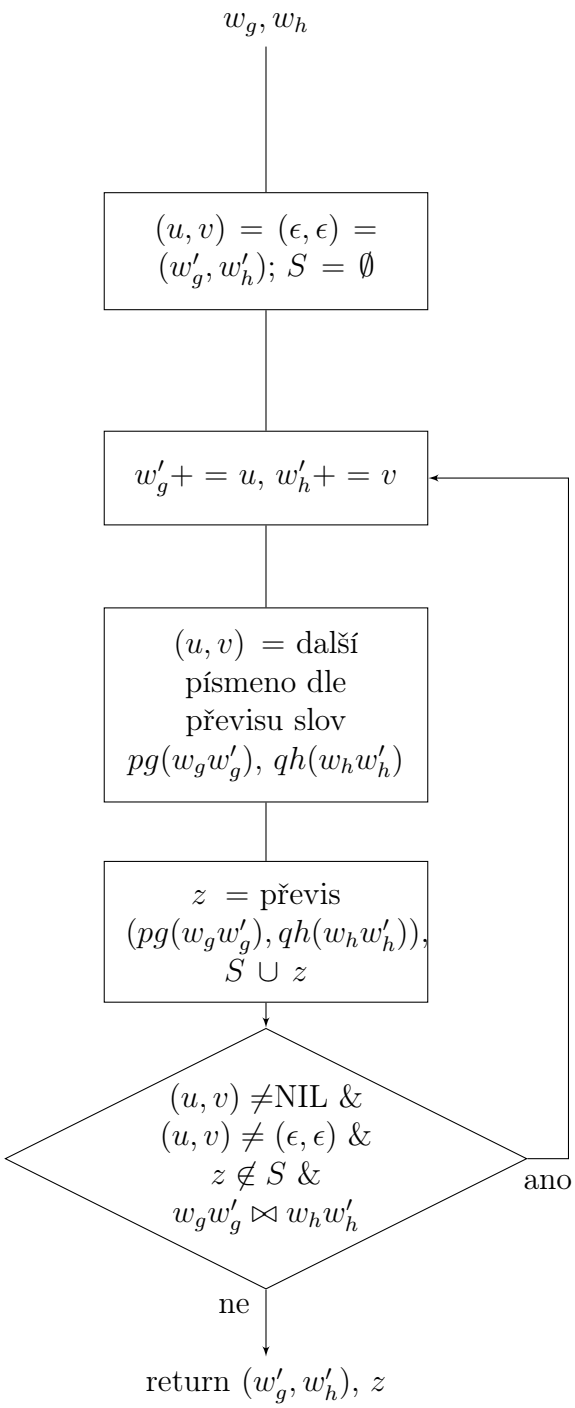
[5] Holub, Štěpán, *Binary morphisms with stable suffix complexity*, International Journal of Foundations of Computer Science, World Scientific Publishing Company, 2010

[6] Choffrut, Christian, Karhumäki, Juhani, *Combinatorics of Words*, In G. Rozenberg and A. Salomaa, editors, Handbook of Formal Languages, volume 1, pp. 329–438, Springer, 1997

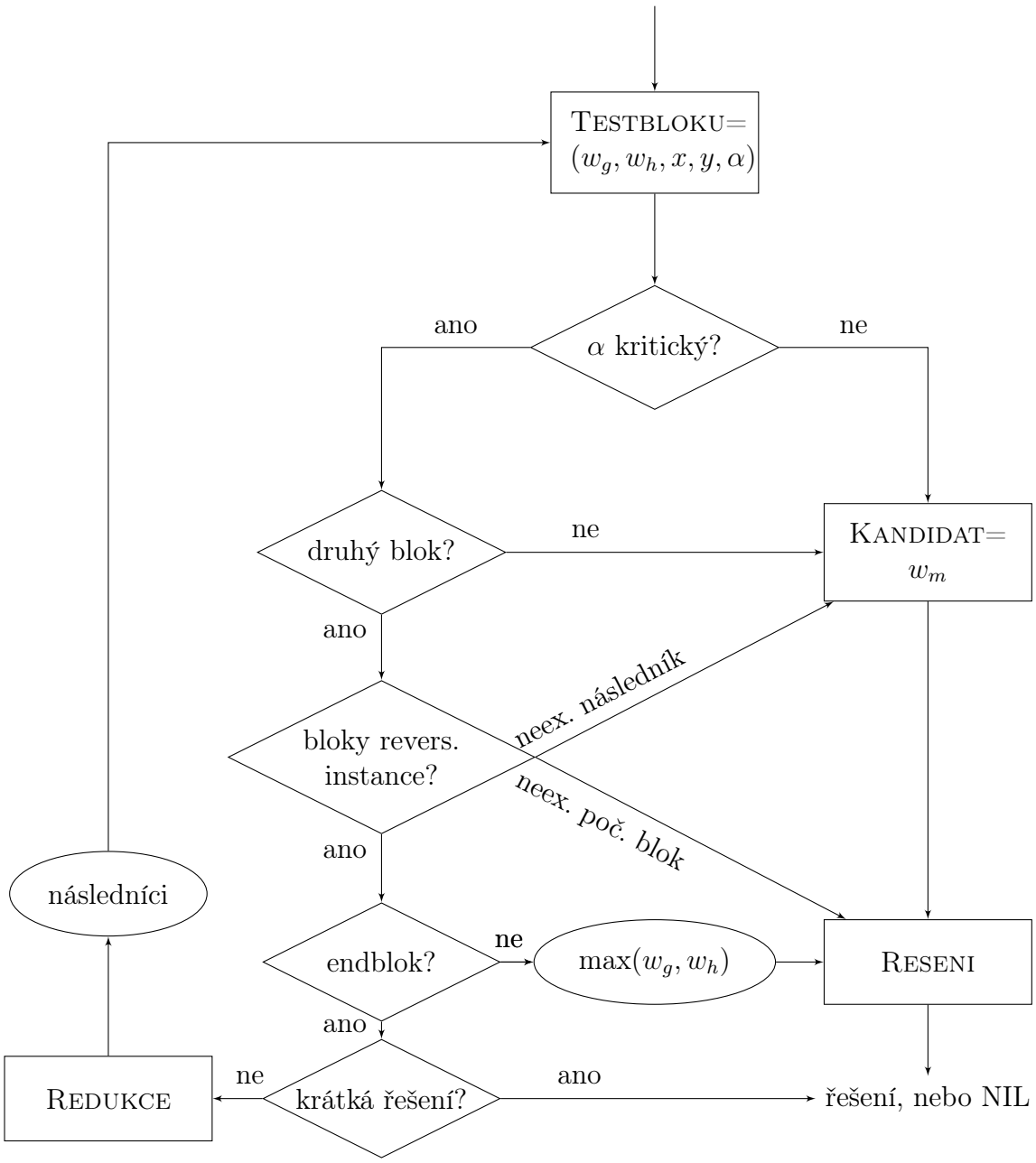
[7] Karhumäki, Juhani, *Combinatorics on Words: a new challenging topic*, Turku: Turku Centre for Computer Science, 2004. ISBN 95-212-1469-4.

[8] Post, Emil, *A variant of a recursively unsolvable problem*, Bull. of Amer. Math. Soc., 52:264–268, 1946

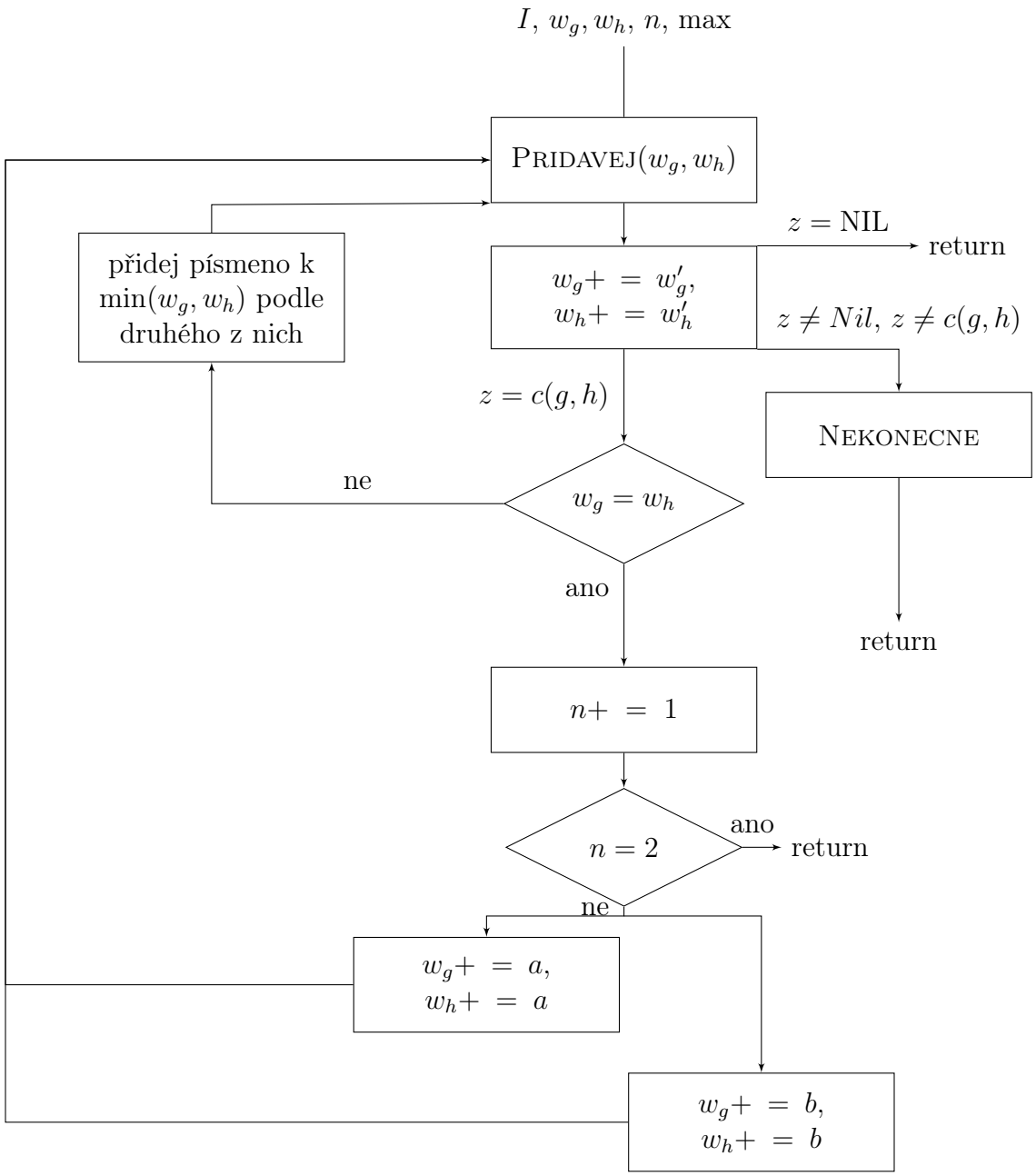
Přílohy



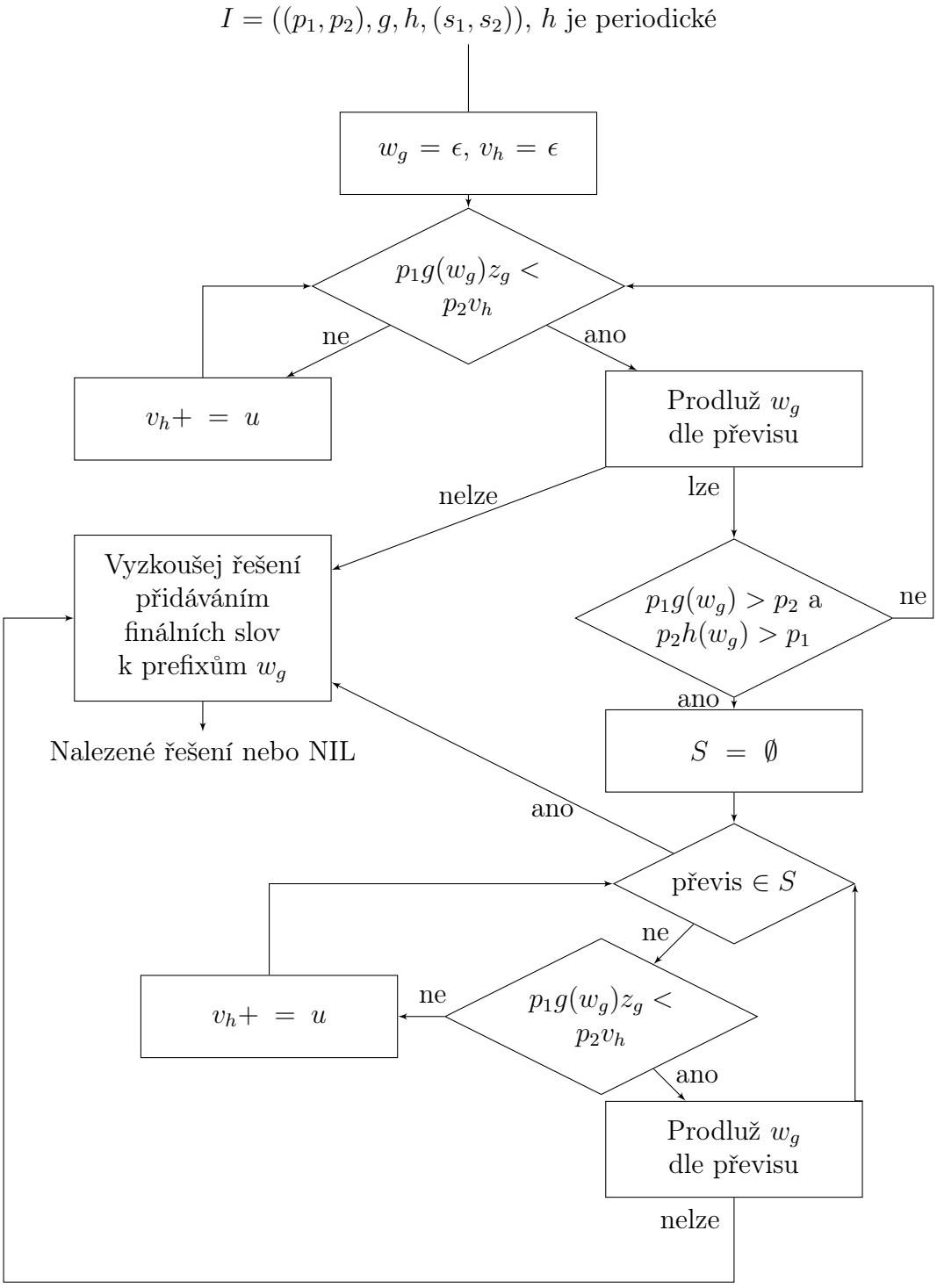
Obrázek 6.1: Funkce PRIDAVEJ



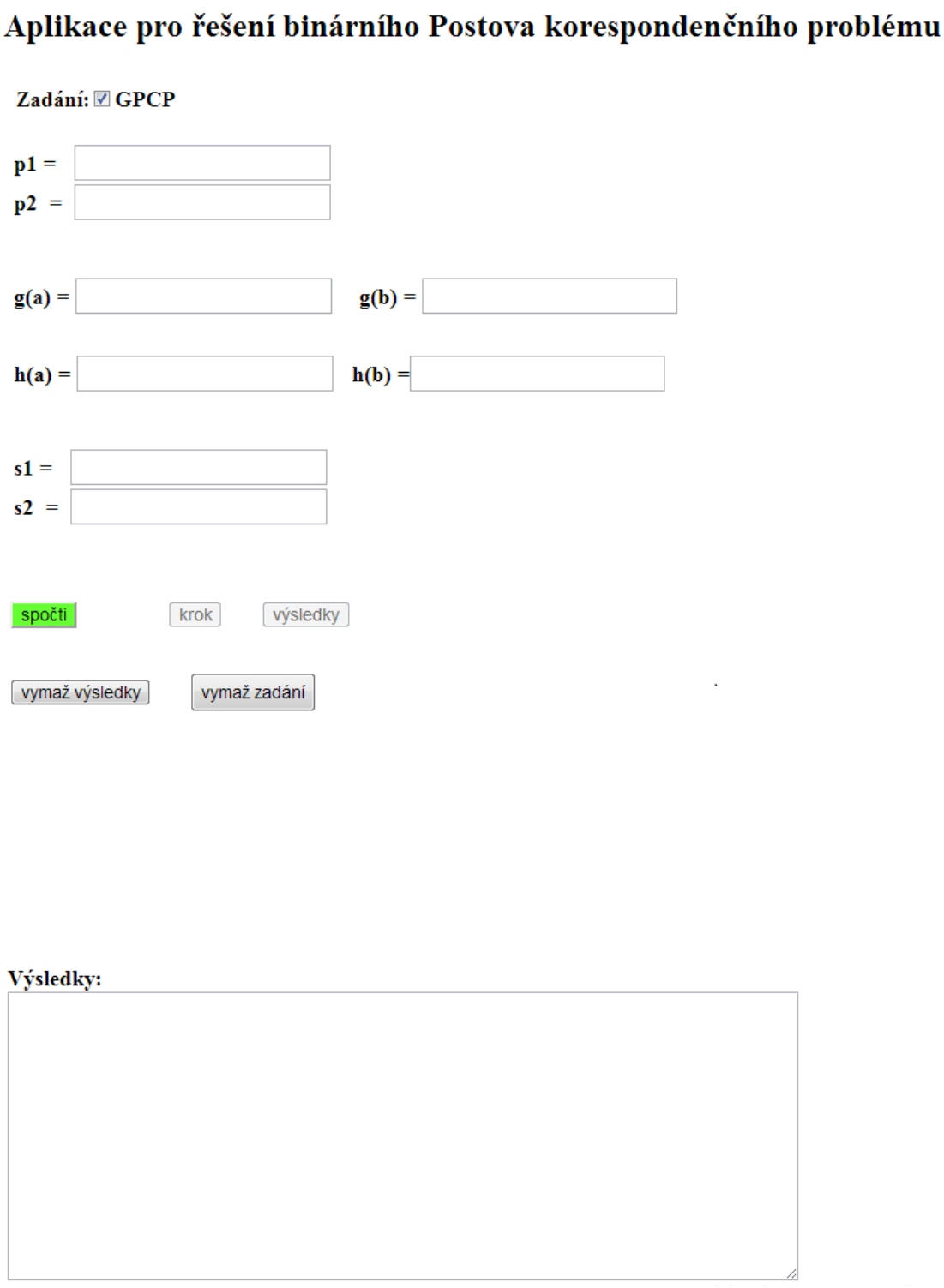
Obrázek 6.2: Blokové schema průběhu algoritmu pro neperiodickou instanci



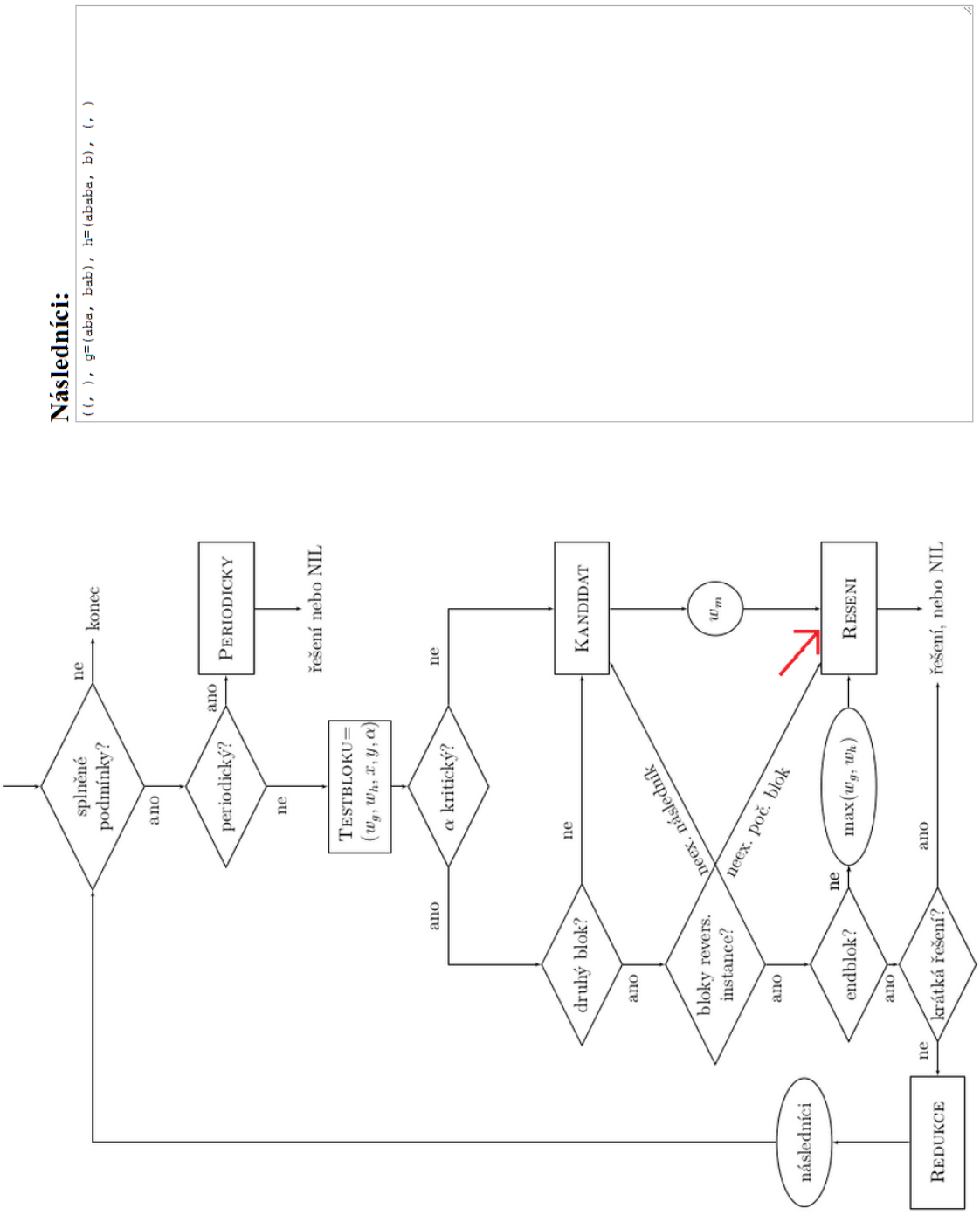
Obrázek 6.3: Funkce KANDIDAT



Obrázek 6.4: Řešení instance s právě jedním periodickým homomorfismem



Obrázek 6.5: Vzhled levé části aplikace - zadávání dat a výpis postupných výsledku



Obrázek 6.6: Vzhled pravé části aplikace - krokování algortimu, výpis posloupnosti následníků