Postovy a konečné stroje

Velmi lehké intro...

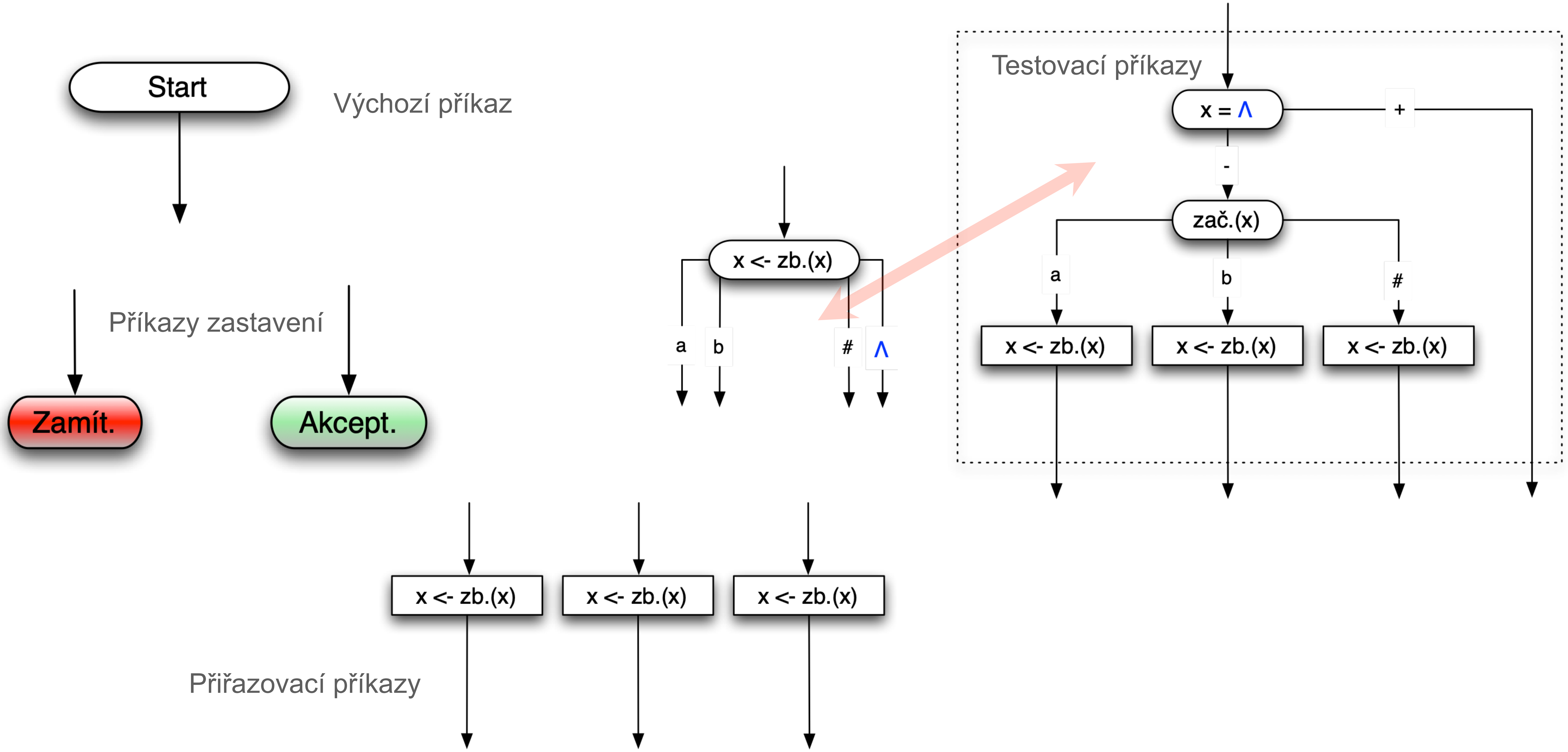
Postovy stroje

Postův stroj je vývojový diagram s jedinou proměnnou x typu fronta (FIFO). Pracuje nad abecedou $\Sigma \cup \{\#\}$. Vývojový diagram může obsahovat následující bloky:

- počáteční blok „START”, koncové bloky „AKCEPT” a „ZAMÍT”
- bloky příkazu přiřazení – příkaz přiřazení přidává symbol na konec proměnné x (přidává zprava)
- blok logického příkazu – logický příkaz umazává jeden symbol v proměnné x zleva a následně podle hodnoty tohoto symbolu větví program do několika větví

Na počátku činnosti stroje je slovo w zápsáno do proměnné x, program začíná blokem „START”. Slovo w je akceptováno, jestliže se stroj v konečném počtu kroků dostane do stavu „AKCEPT”. Slovo w je zamítnuto jestliže se stroj v konečném počtu kroku dostane do stavu „ZAMÍT”. V ostatních případech stroj cykluje.

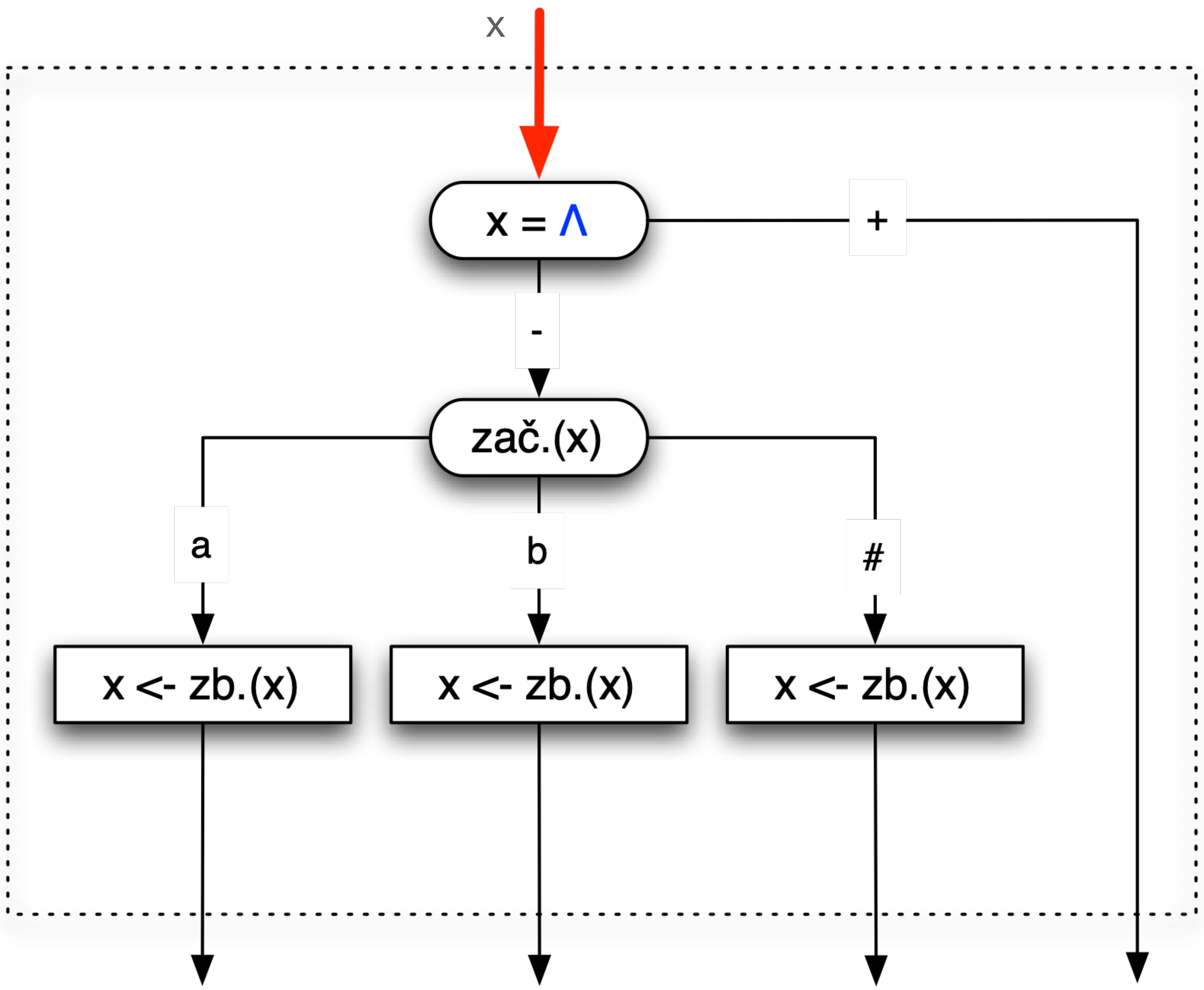
Postovy stroje



Postovy stroje

if x = Λ
then GO +
else
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

x = Λ

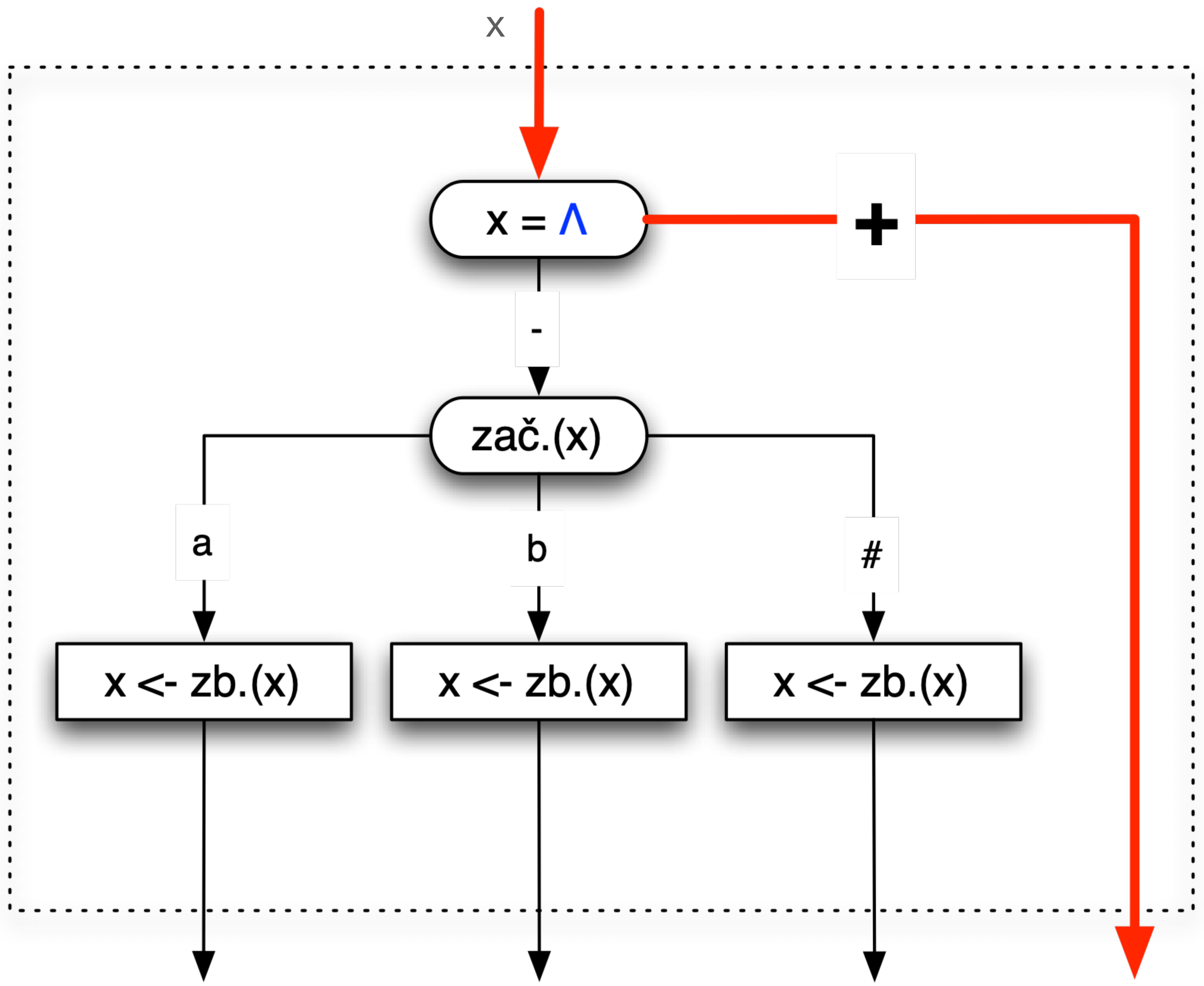


Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

Postovy stroje

if x = Λ
then GO +
else
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

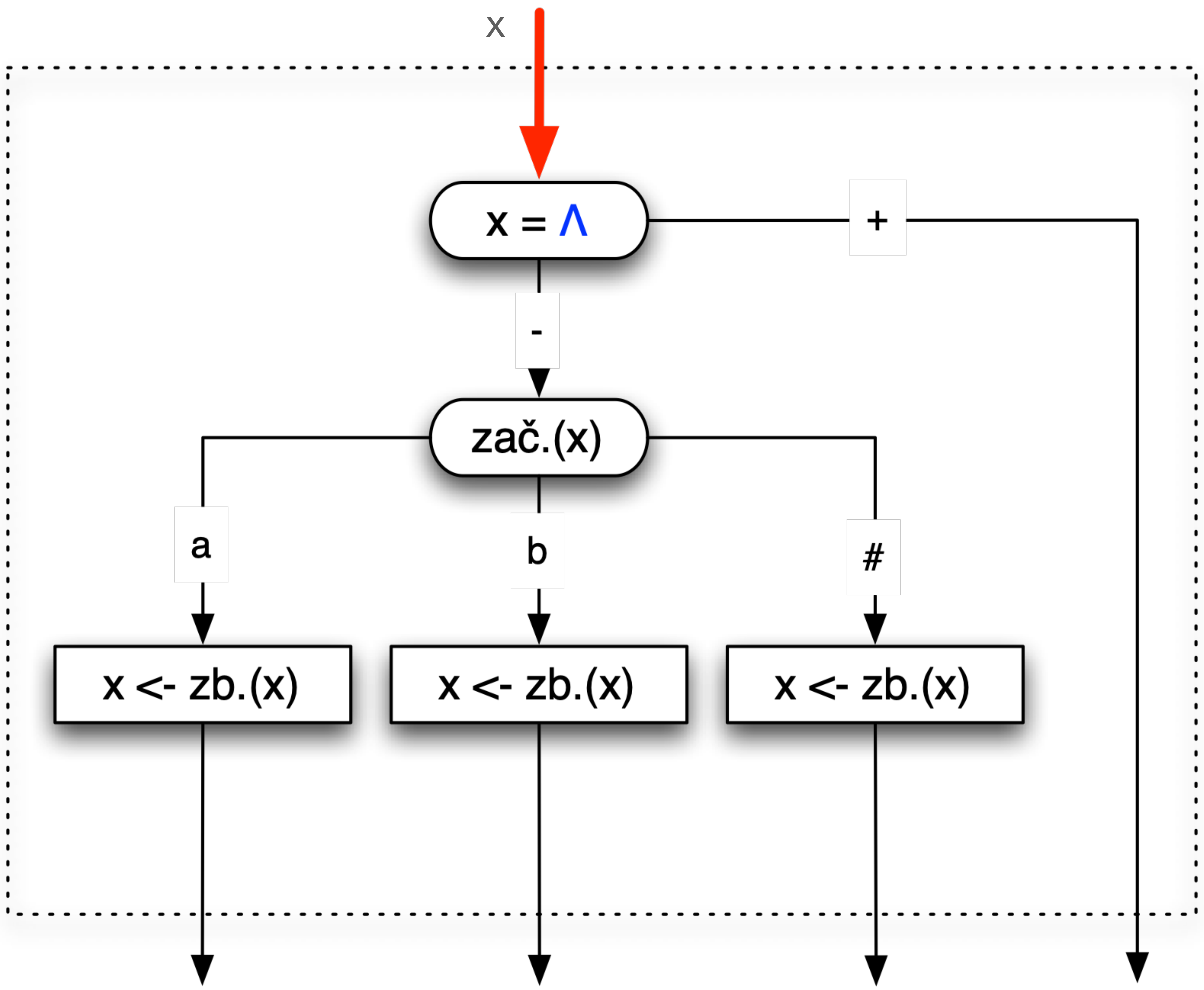
x = Λ



Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

Postovy stroje

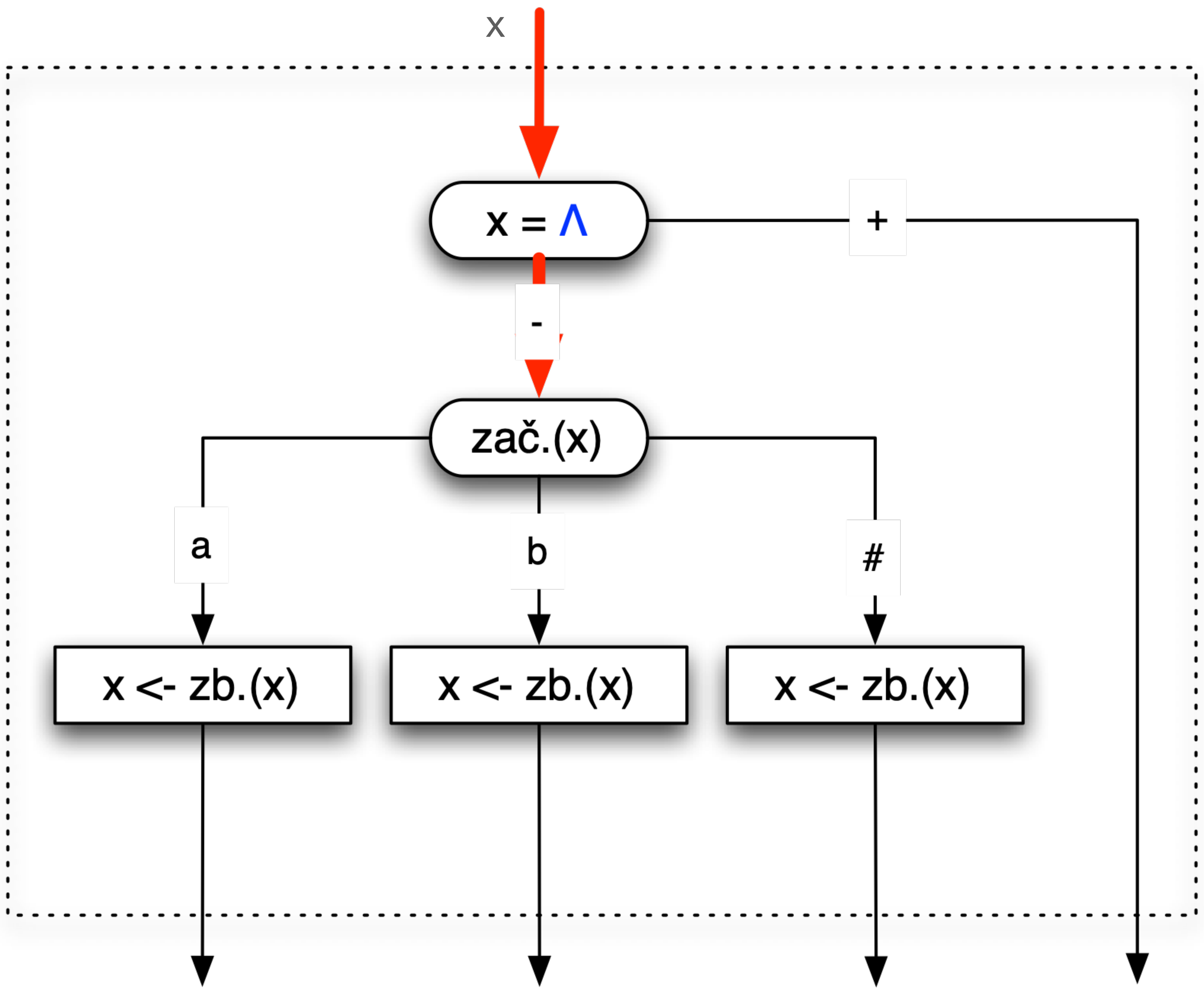
x = #ab



if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

Postovy stroje



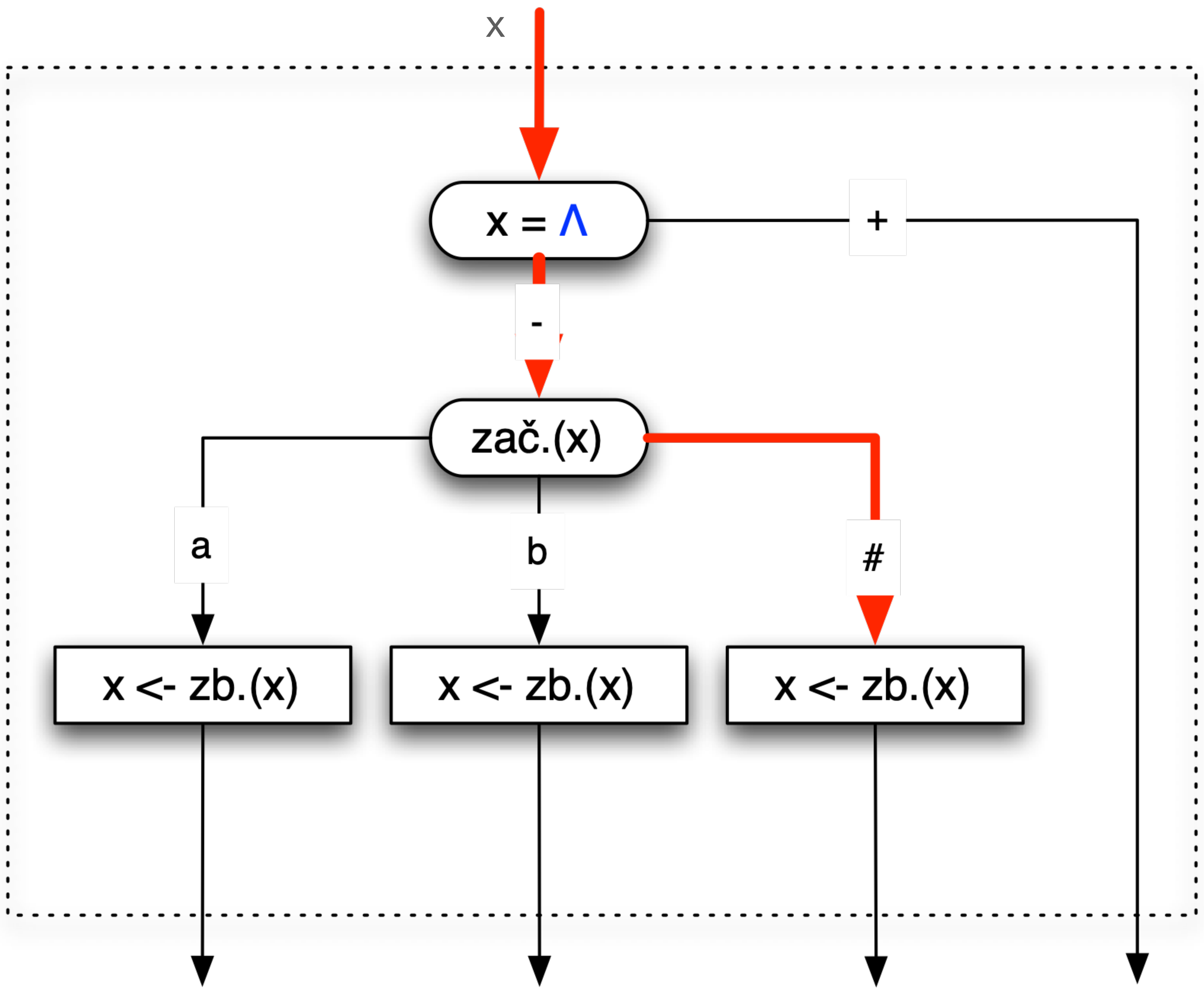
x = #ab

Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

x = #ab



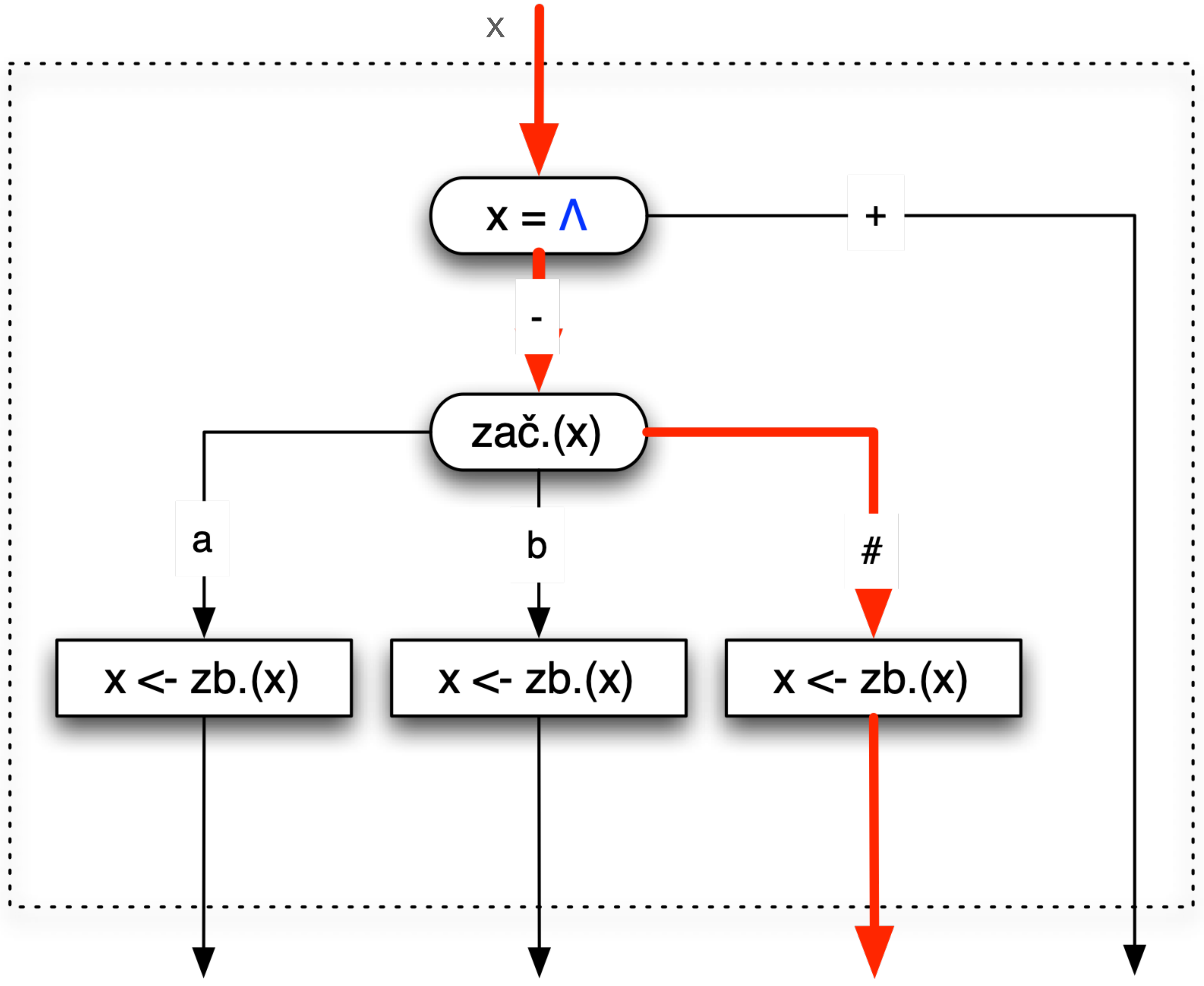
Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

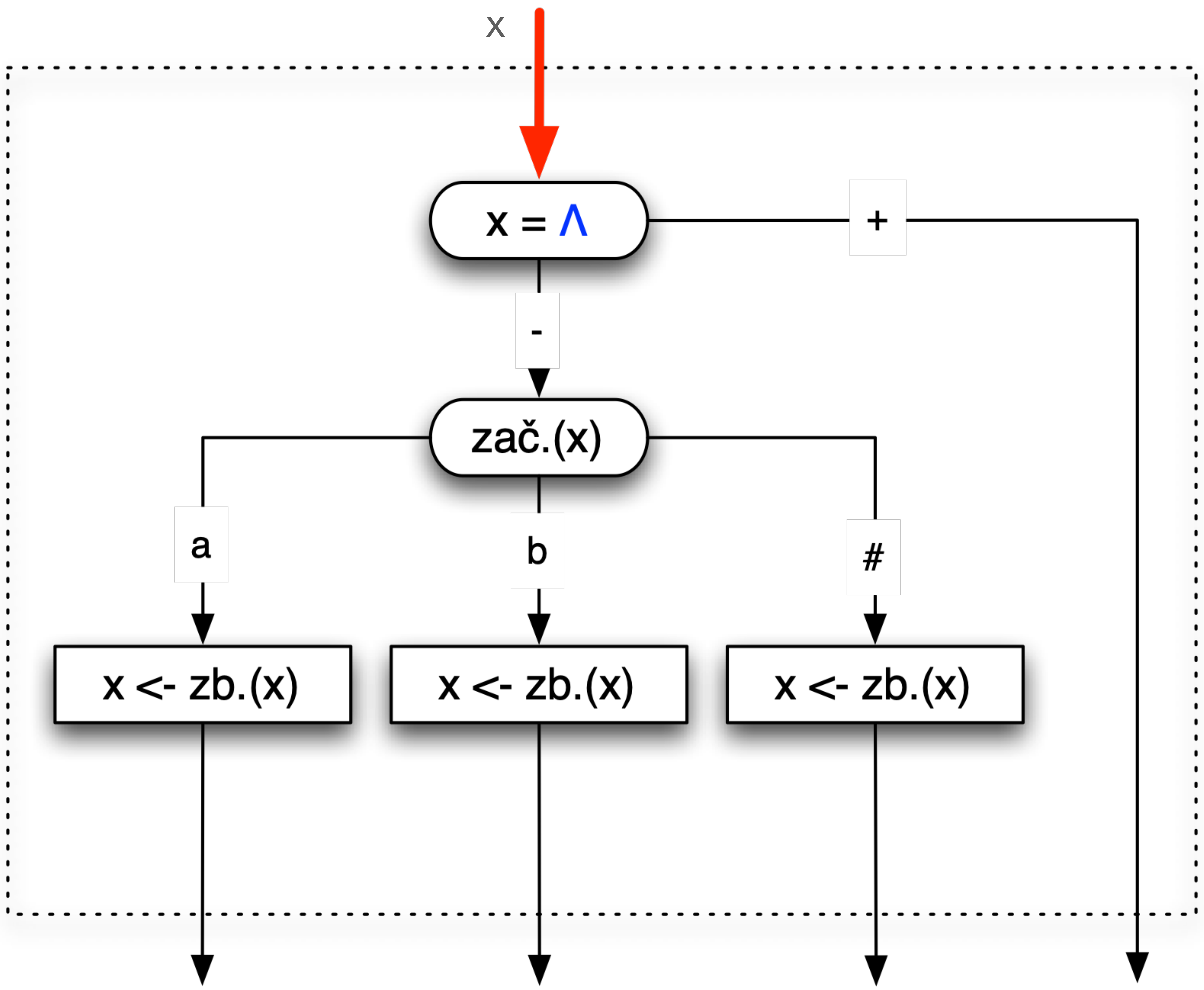
if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

x = ab



Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

Postovy stroje



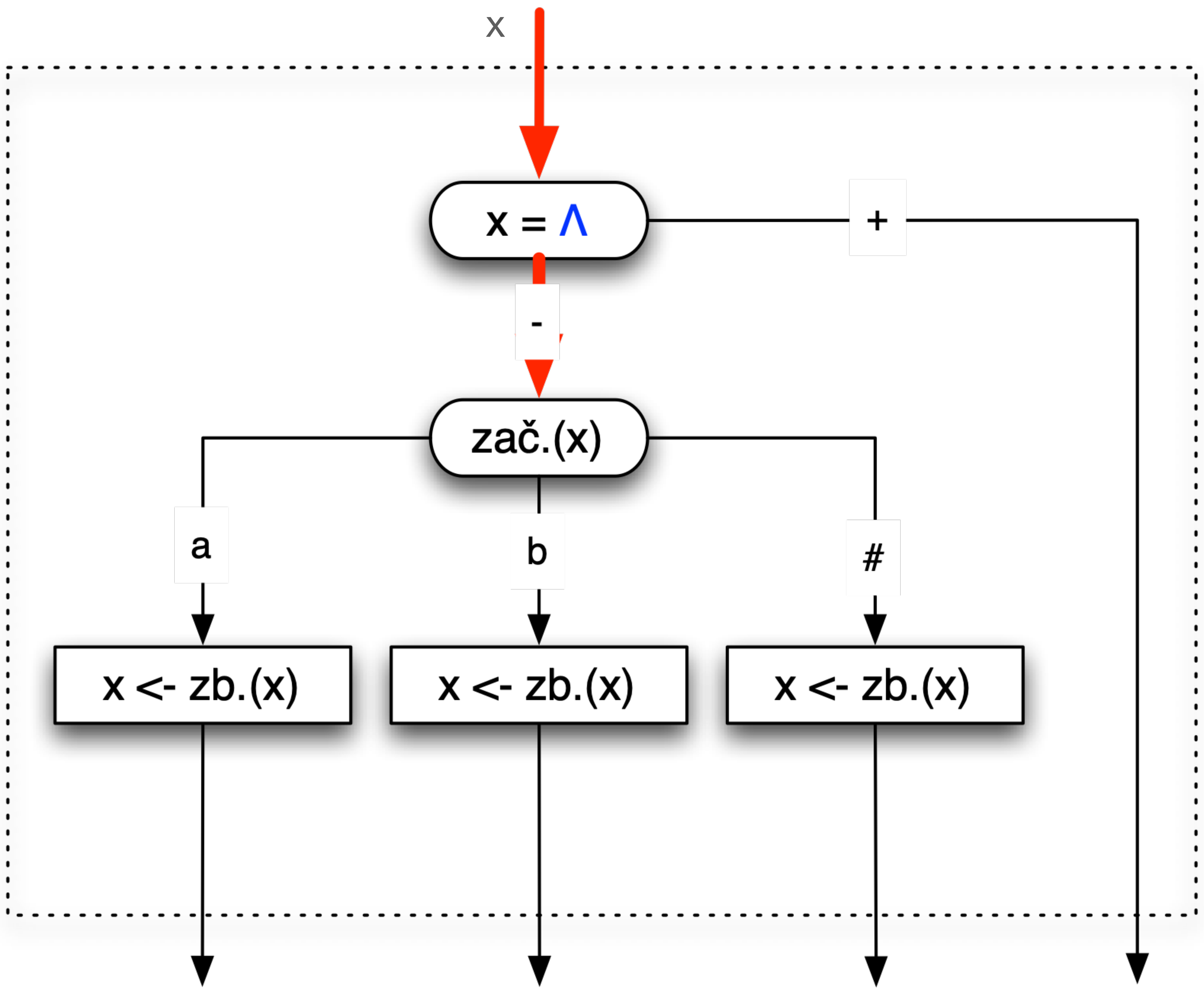
x = ba#

Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

```
if x = Λ
then GO +
else GO -
If zač(x) = #
    GO #
    x = ab
else if zač(x)=b
    GO b
    x = a#
else if zač(x)=a
    GO a
    x = b#
   end
end
```

Postovy stroje

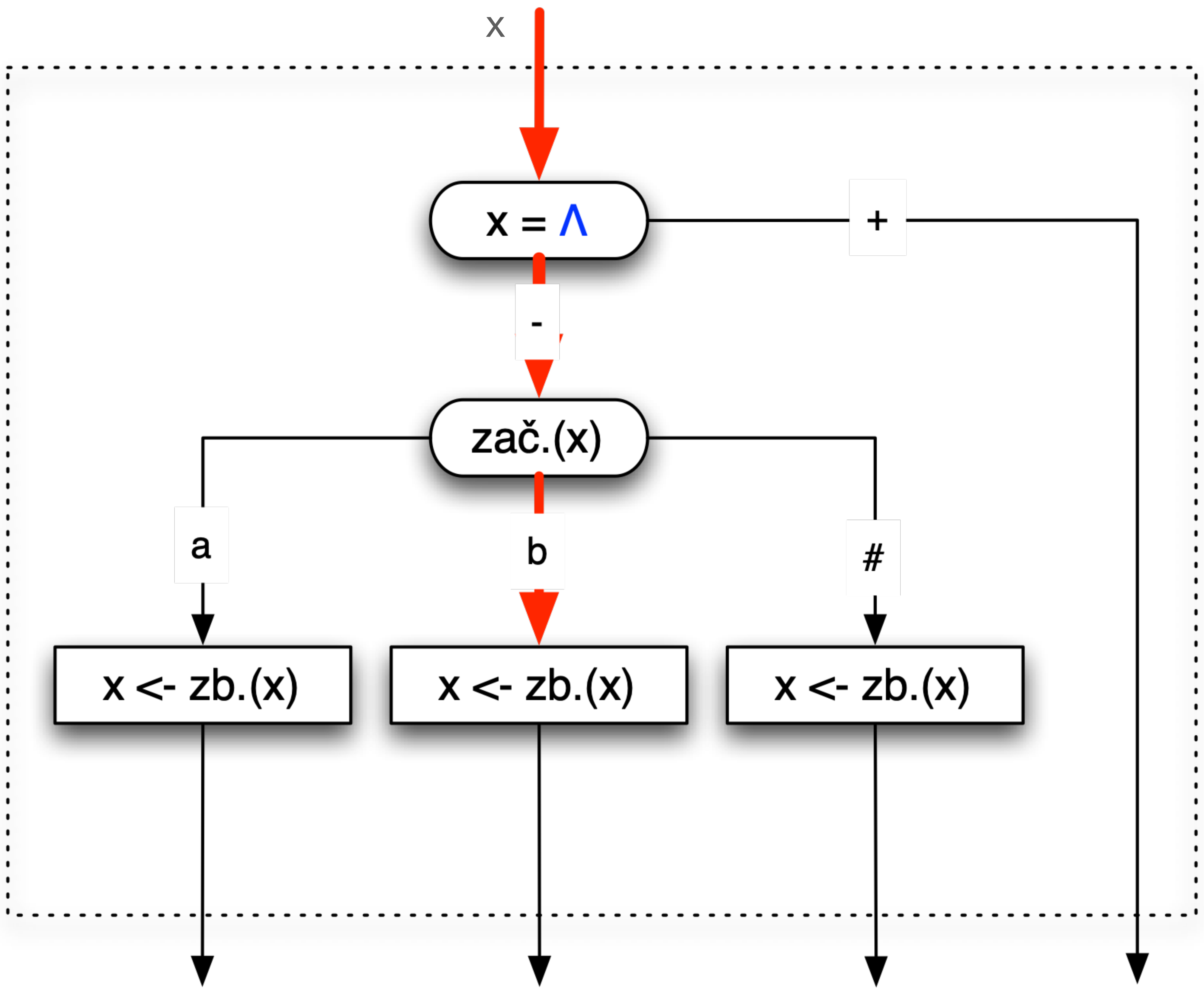
x = ba#



Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje



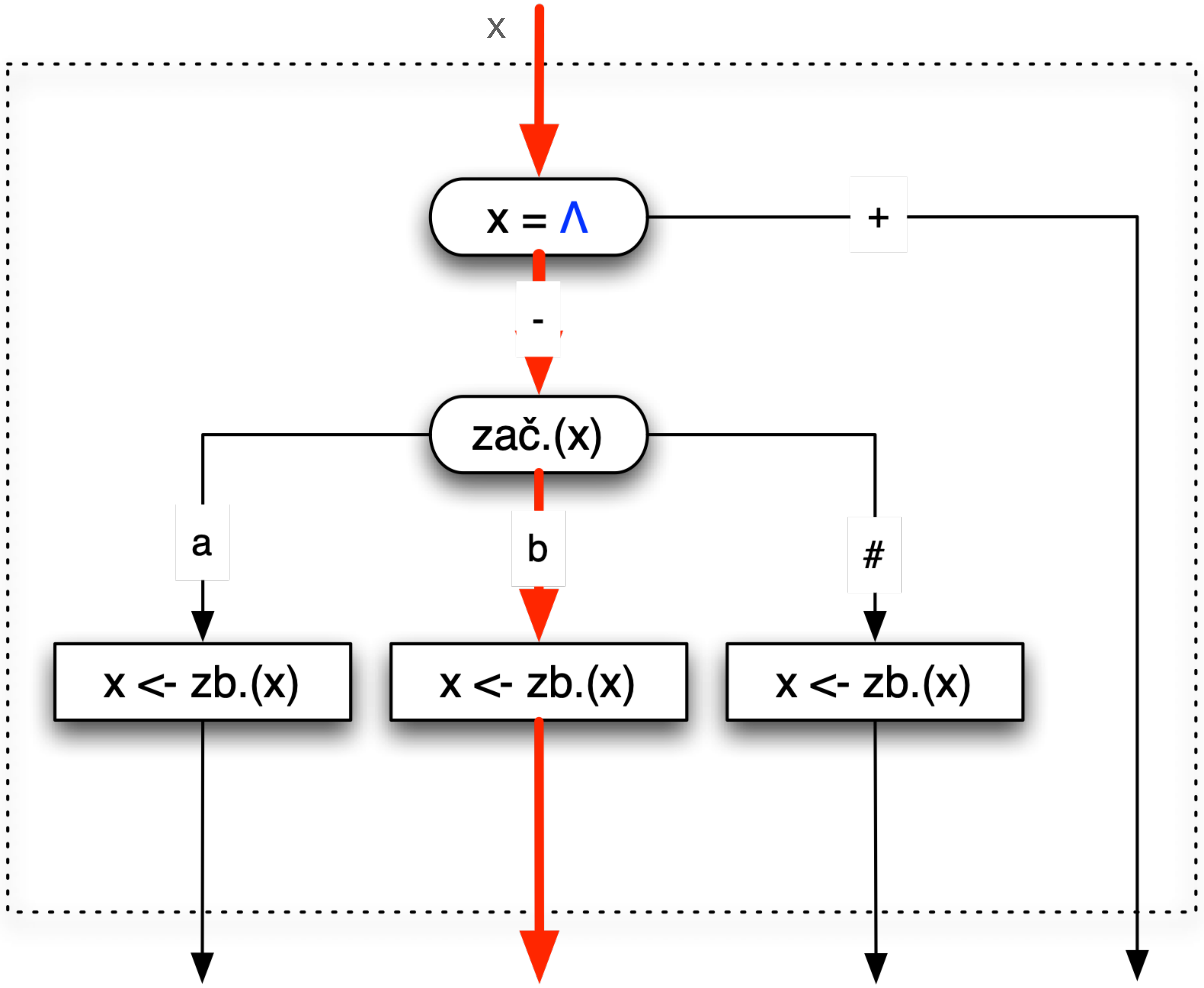
x = ba#

Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

x = a#

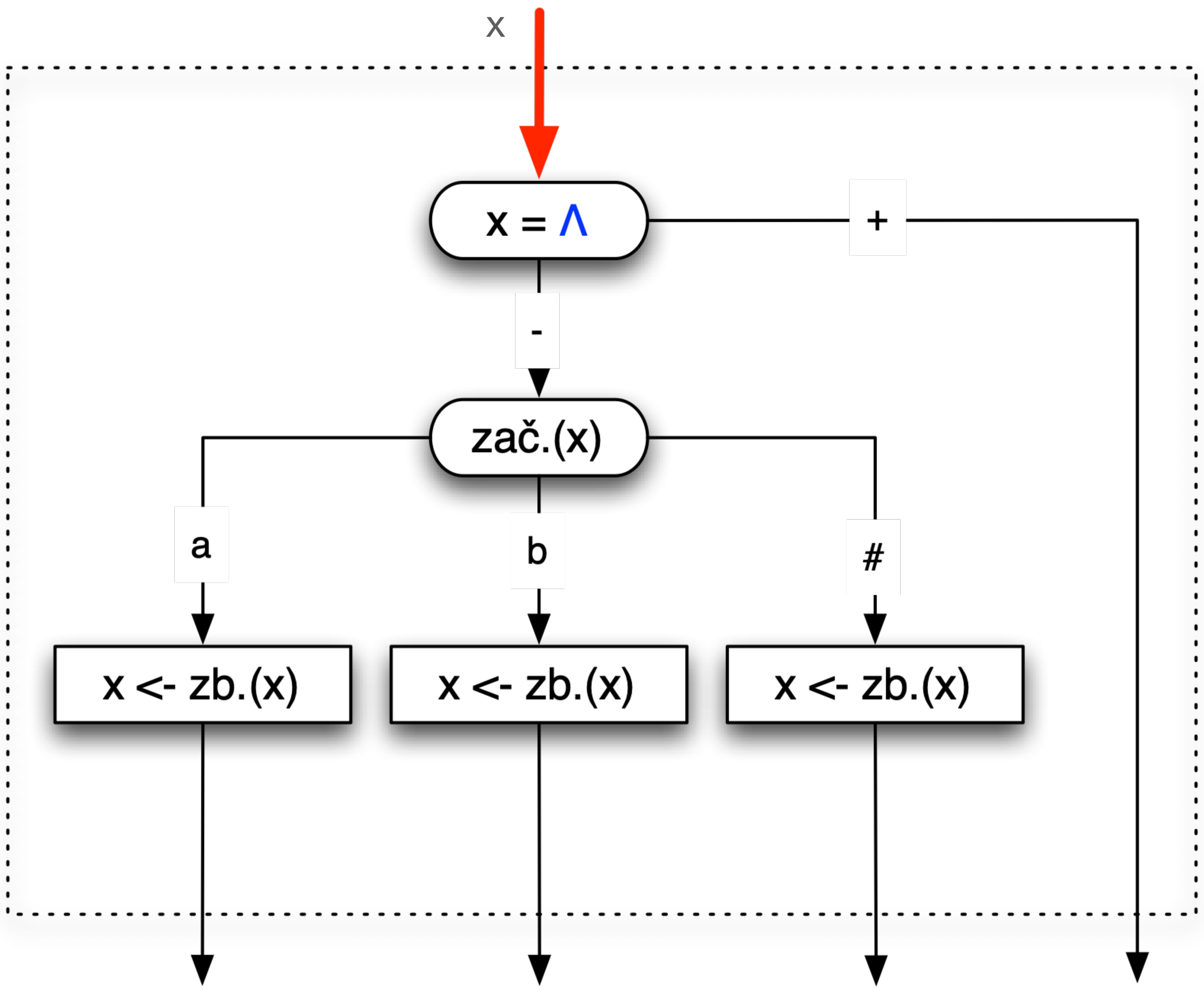


Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

x = ab#

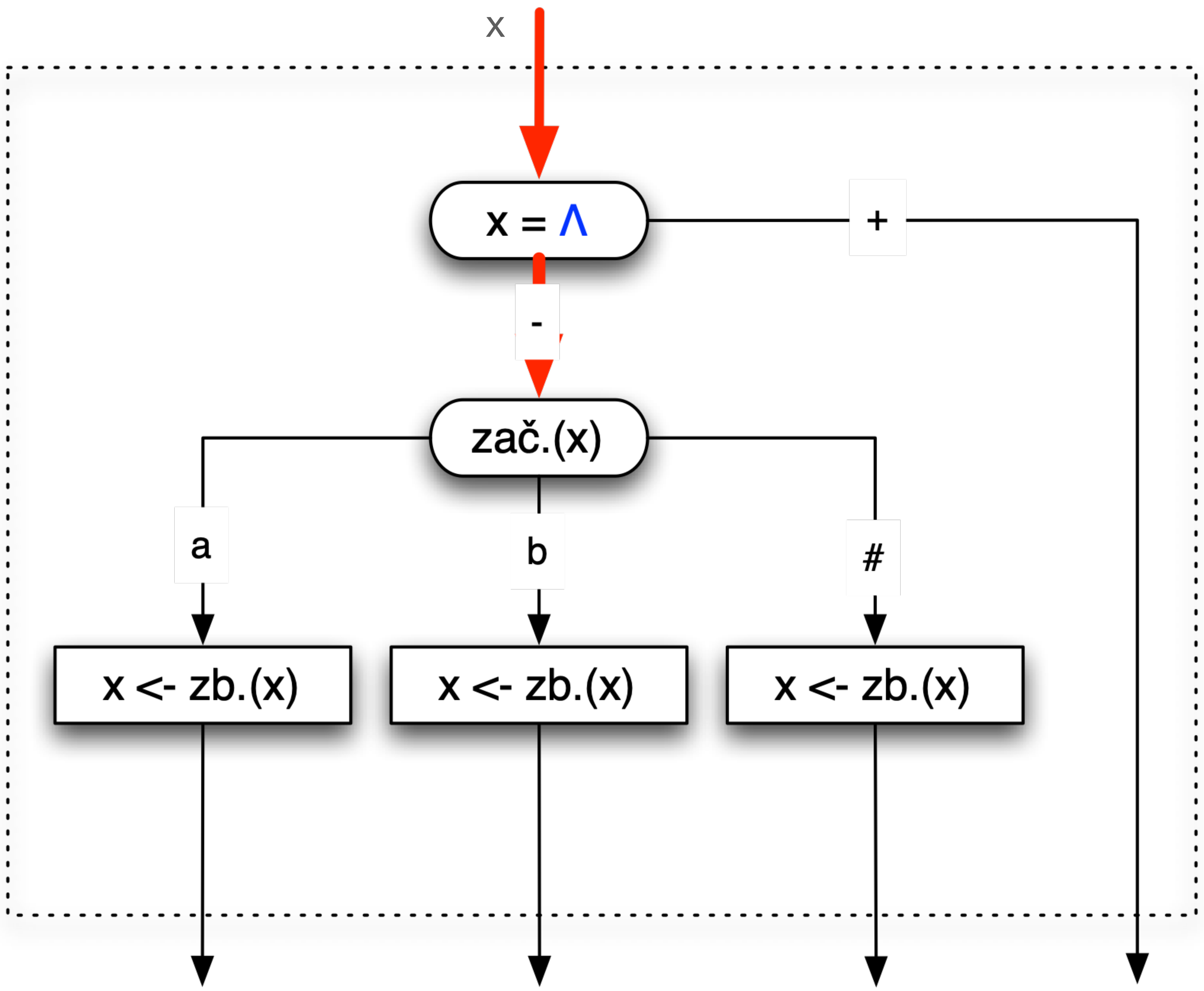


Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

x = ab#

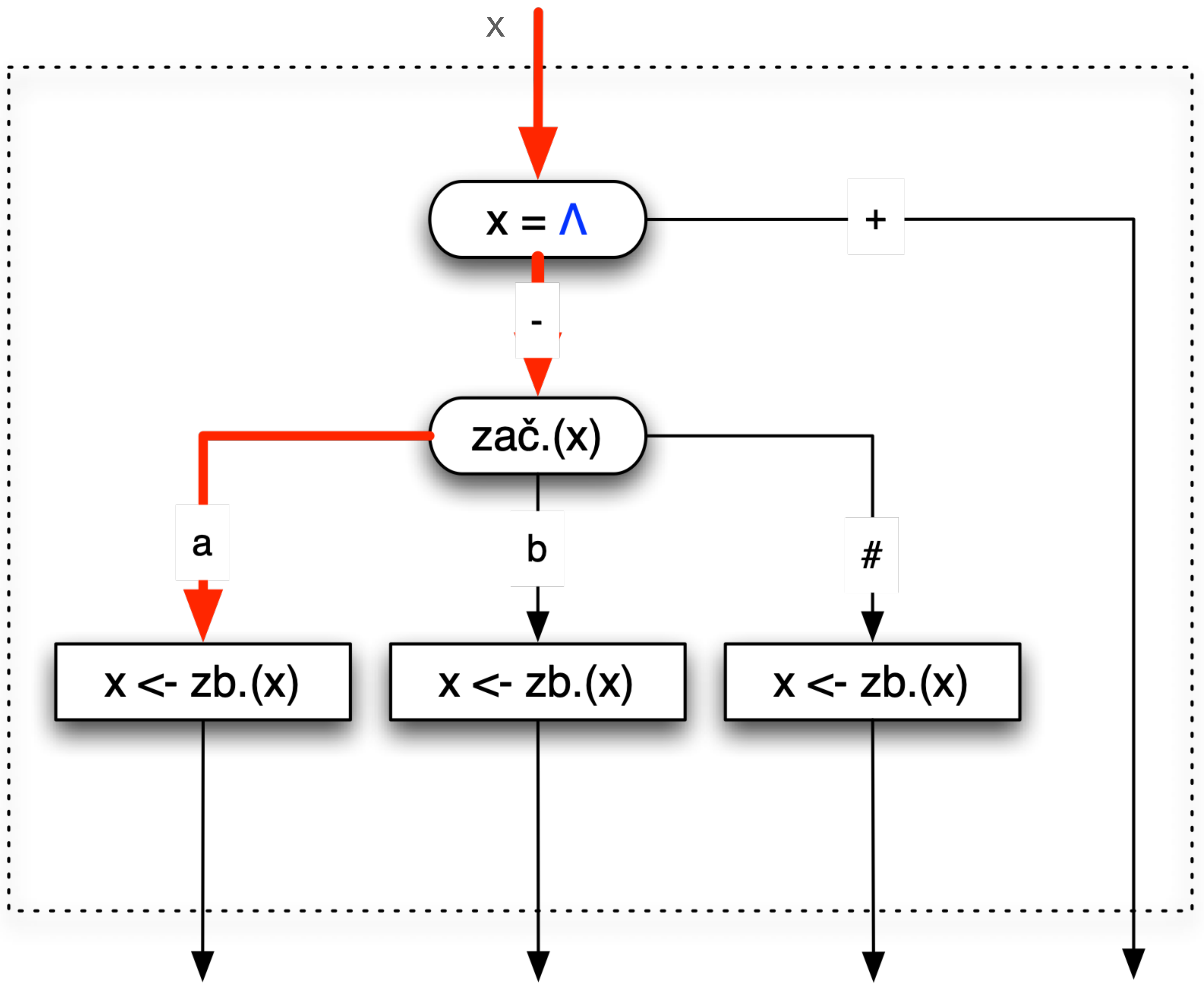


Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Postovy stroje

x = ab#



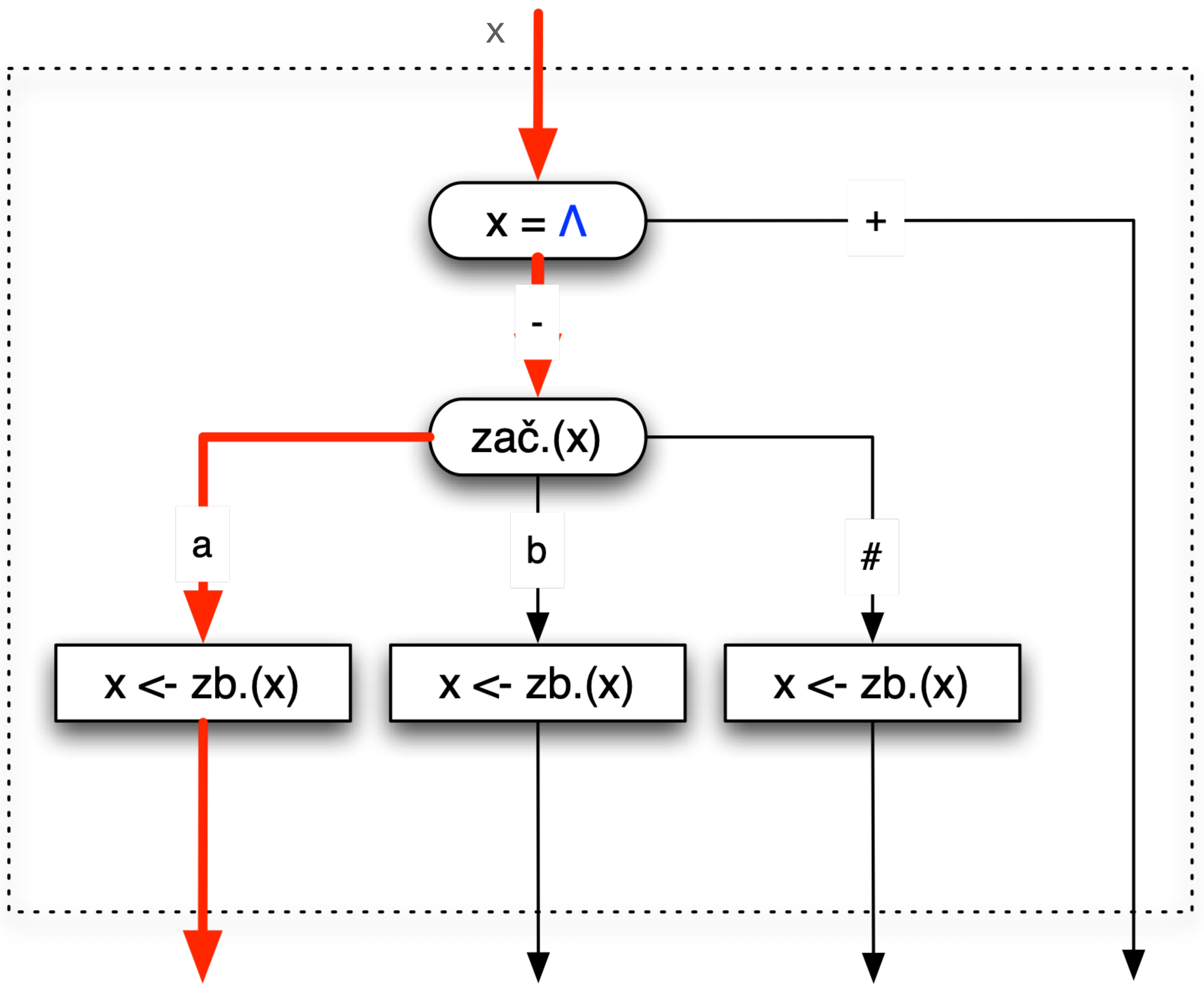
if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

Postovy stroje

if x = Λ
then GO +
else GO -
If zač(x) = #
GO #
x = ab
else if zač(x)=b
GO b
x = a#
else if zač(x)=a
GO a
x = b#
end
end

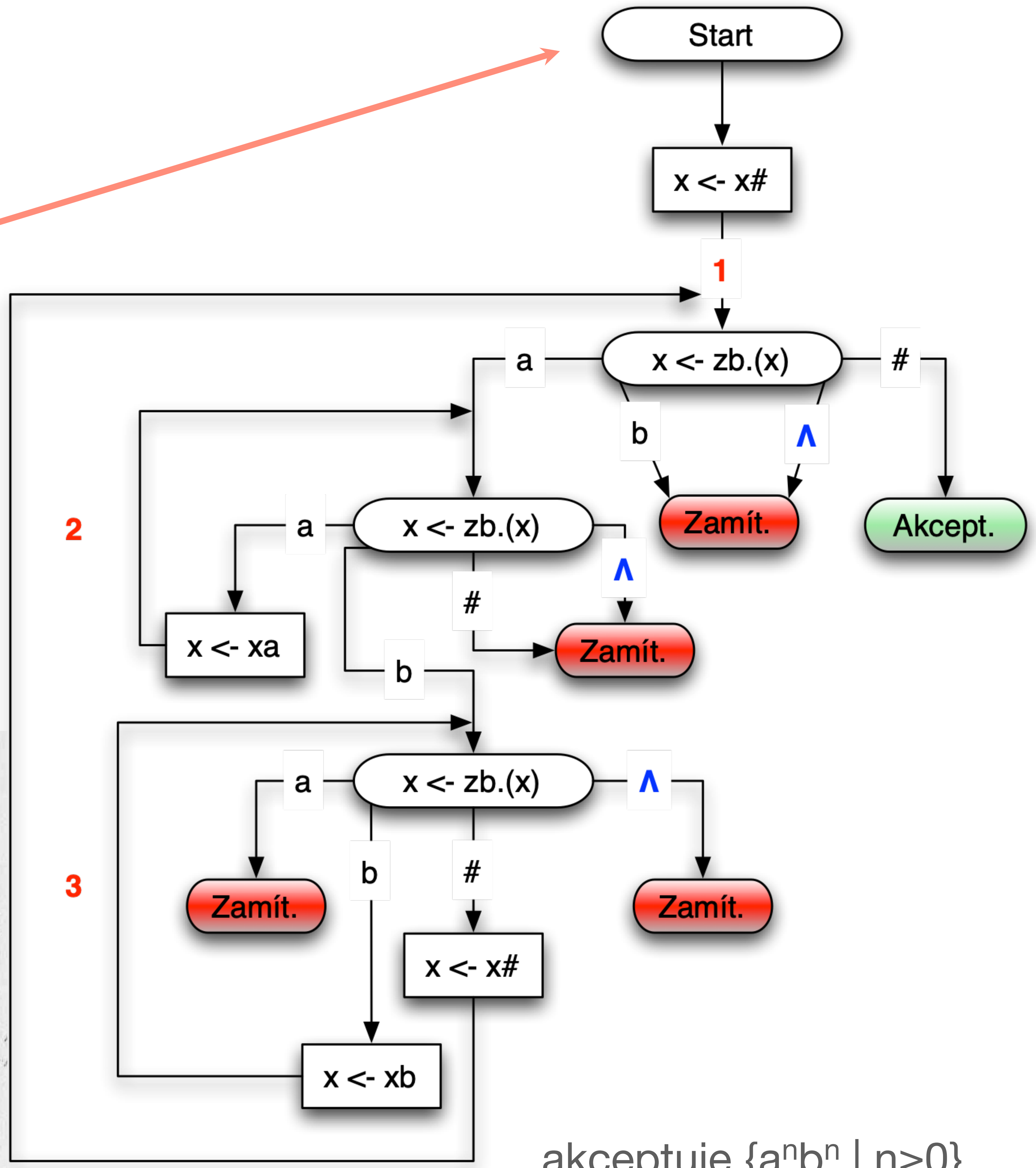
x = b#



Příkazy *GO* v pseudokódu jsou uvedeny jen jako zdůraznění faktu, jakou cestou se činnost stroje (algoritmu) bude ubírat

PS příklad

aabb

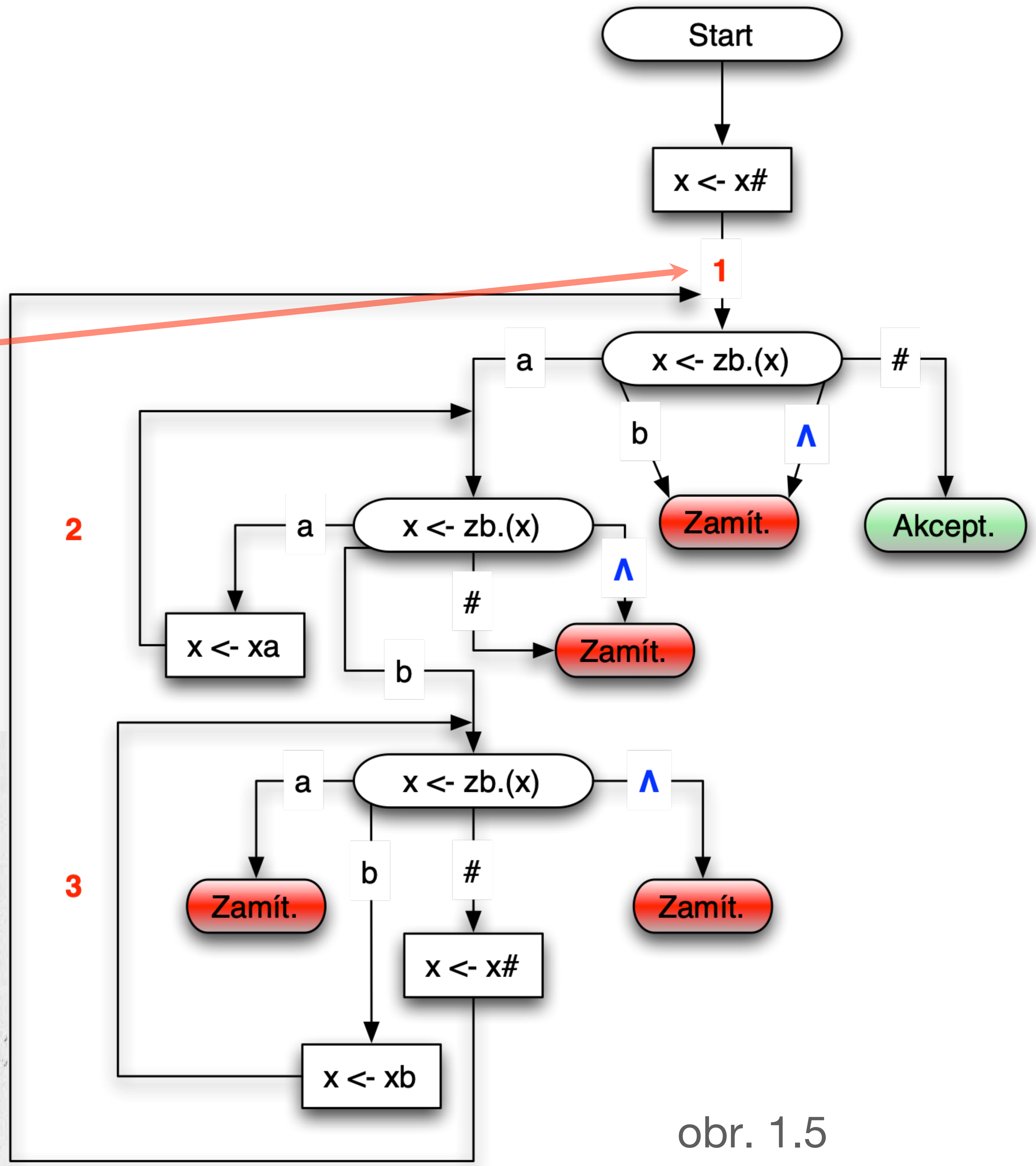


akceptuje $\{a^nb^n \mid n \geq 0\}$

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru a^nb^n, $n \geq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^nb^n | n \geq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^nb^n$ při nějakém pevném $n \geq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^ib^i\#$ pro vhodné i, $0 \leq i \leq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i\#a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\#a^{i-1}b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1}b^{i-1}\#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

PS příklad

aabb#



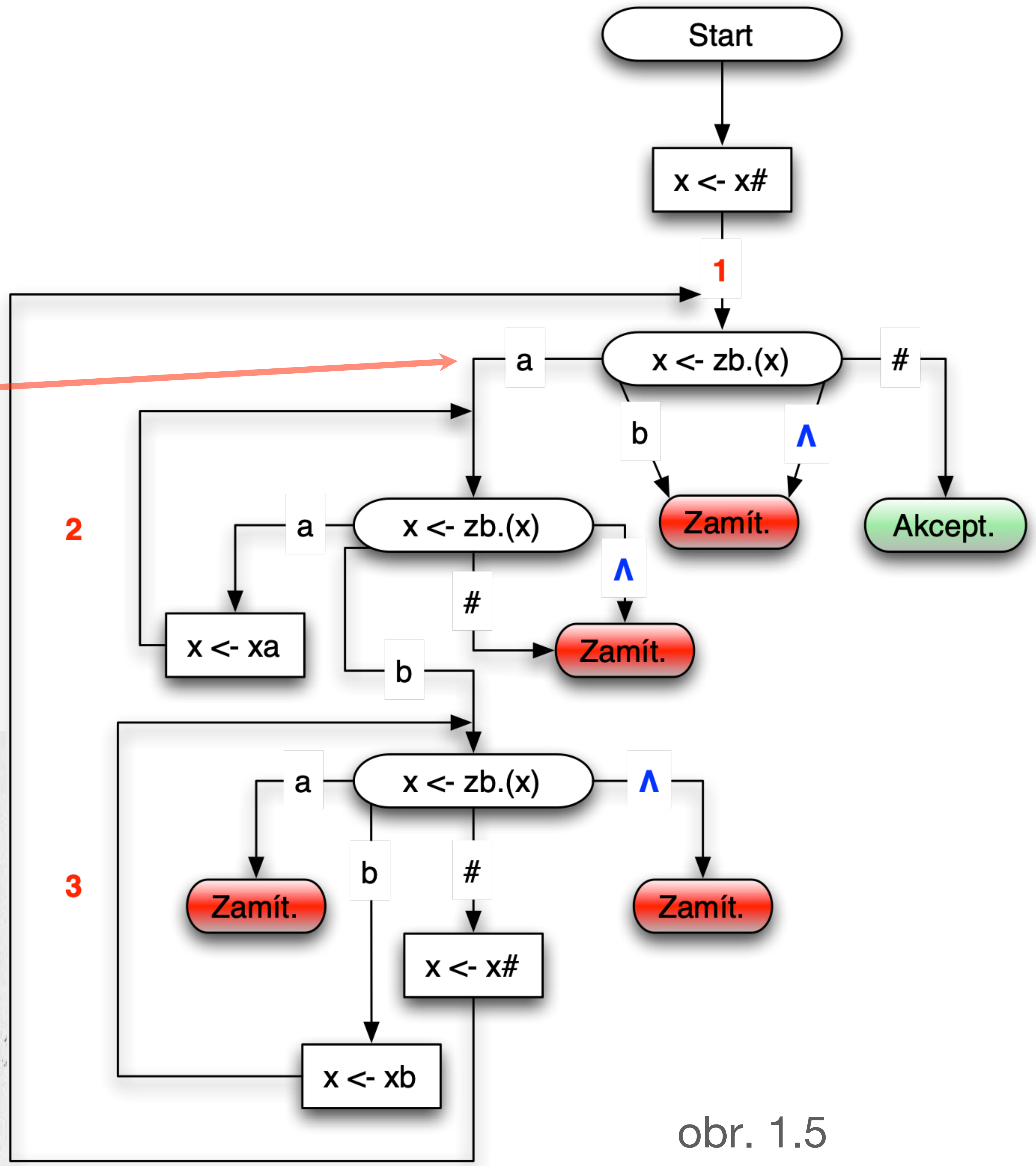
obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leq i \leq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

PS příklad

abb#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol # na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol # a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

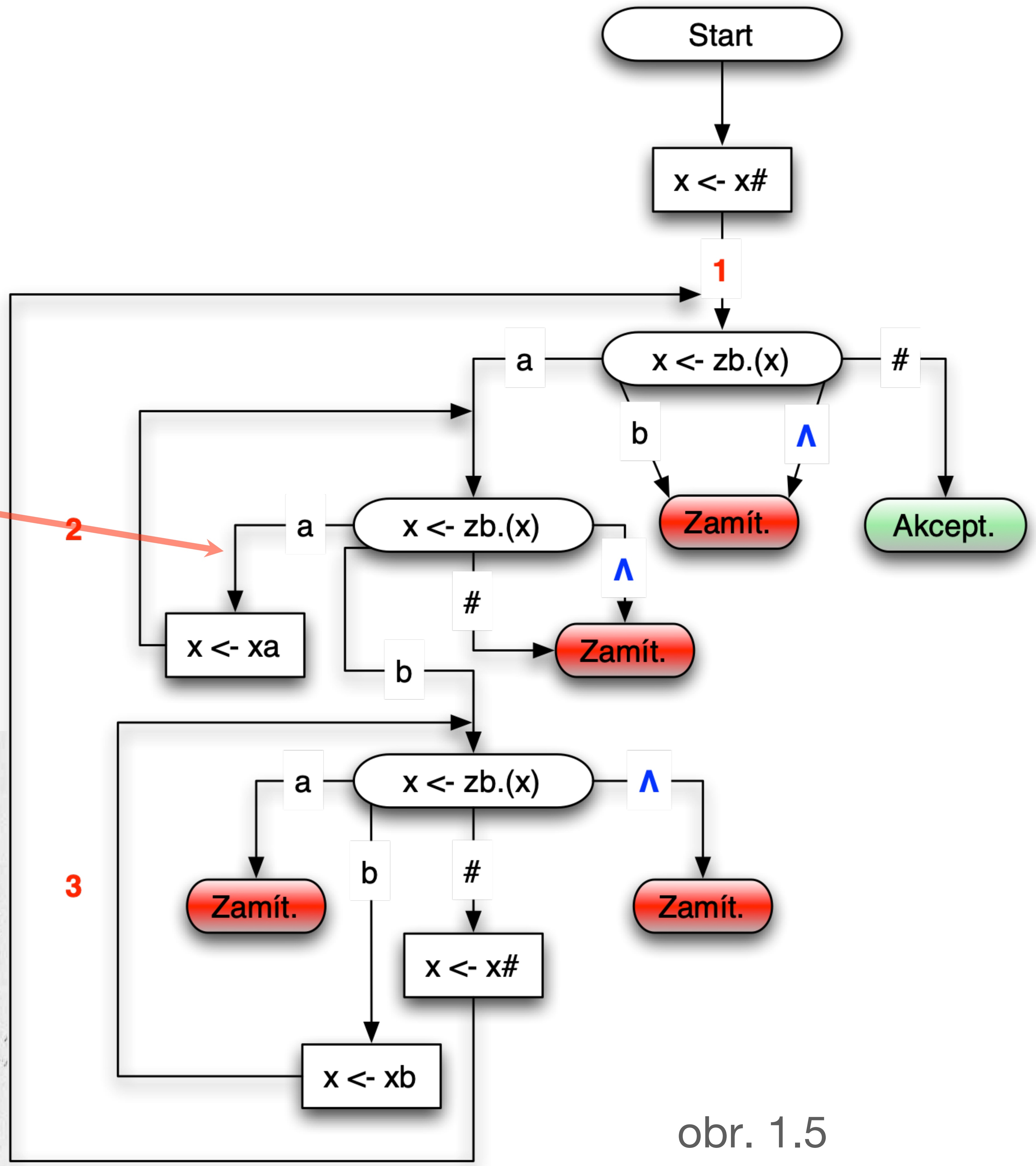


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

bb#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol # na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol # a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

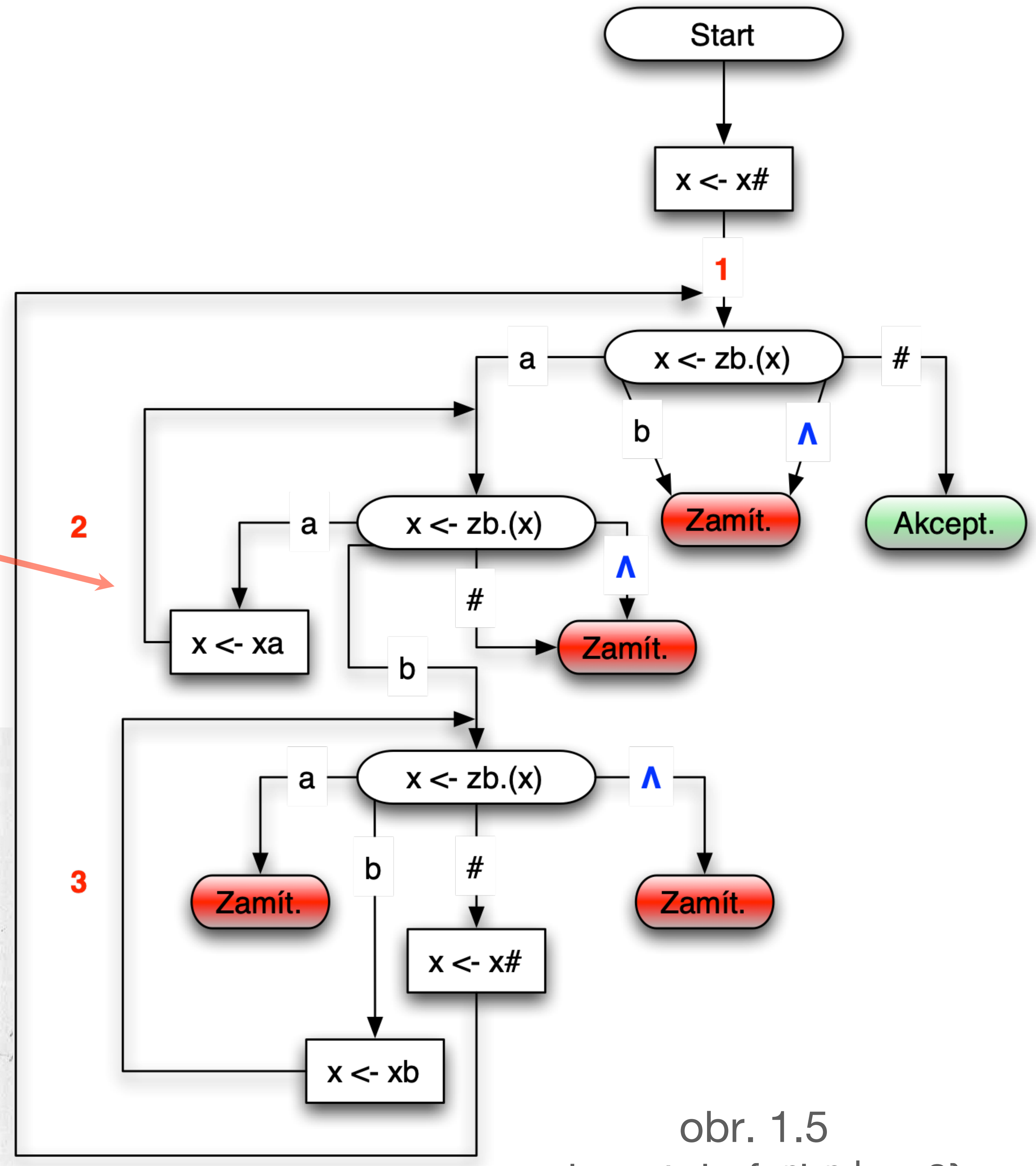


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

bb#a

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

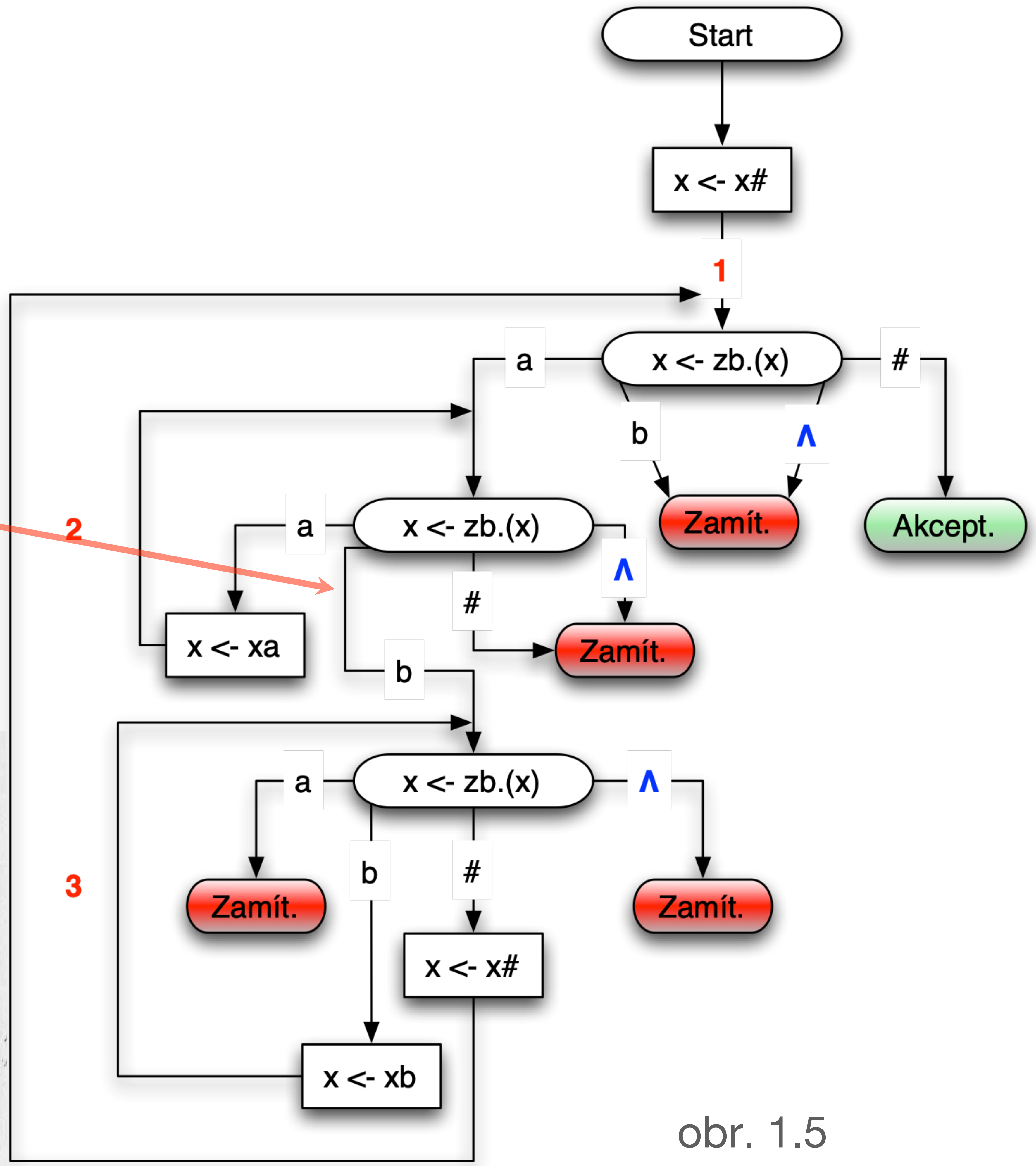


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

b#a

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

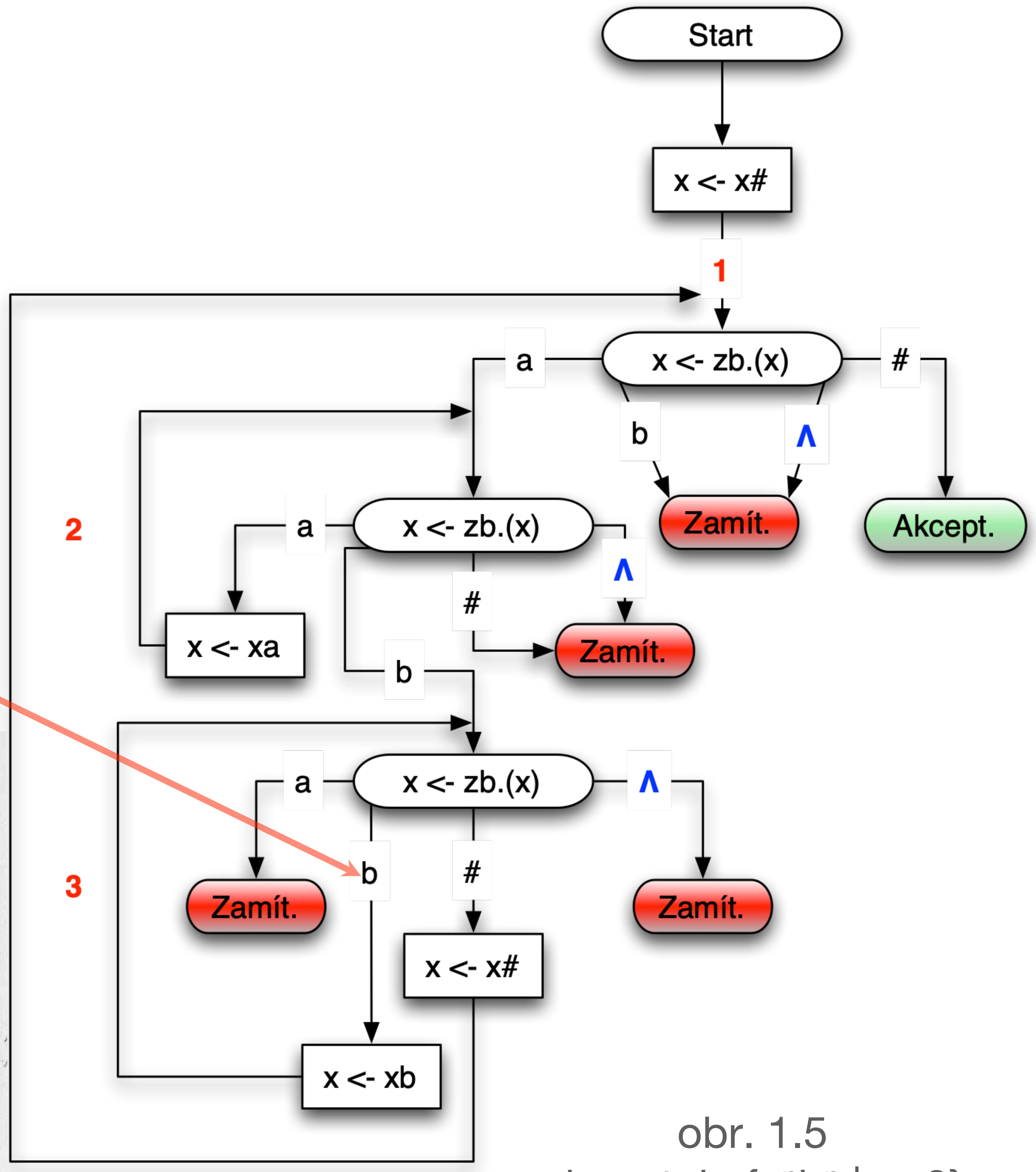


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

#a

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

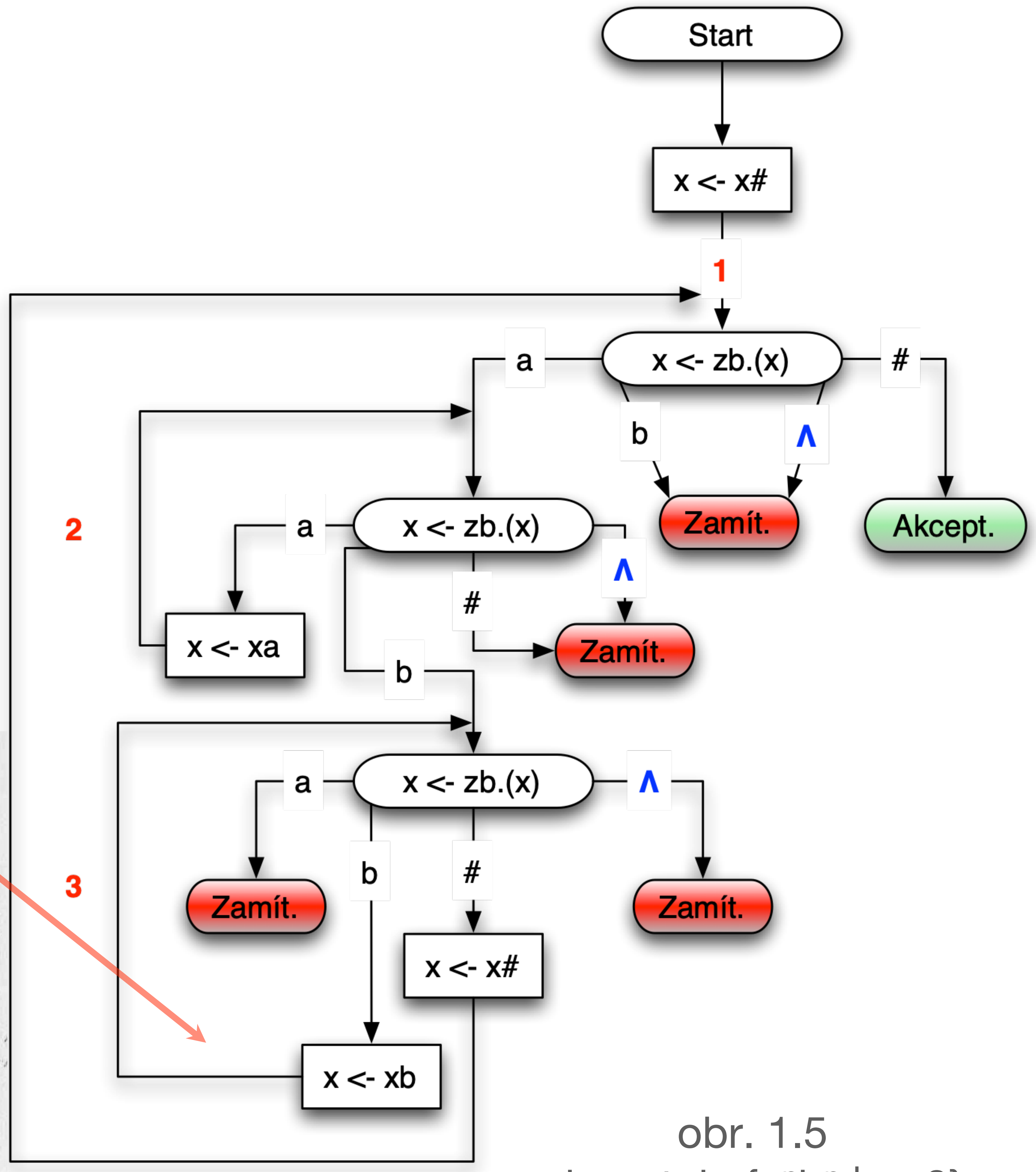


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

#ab

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

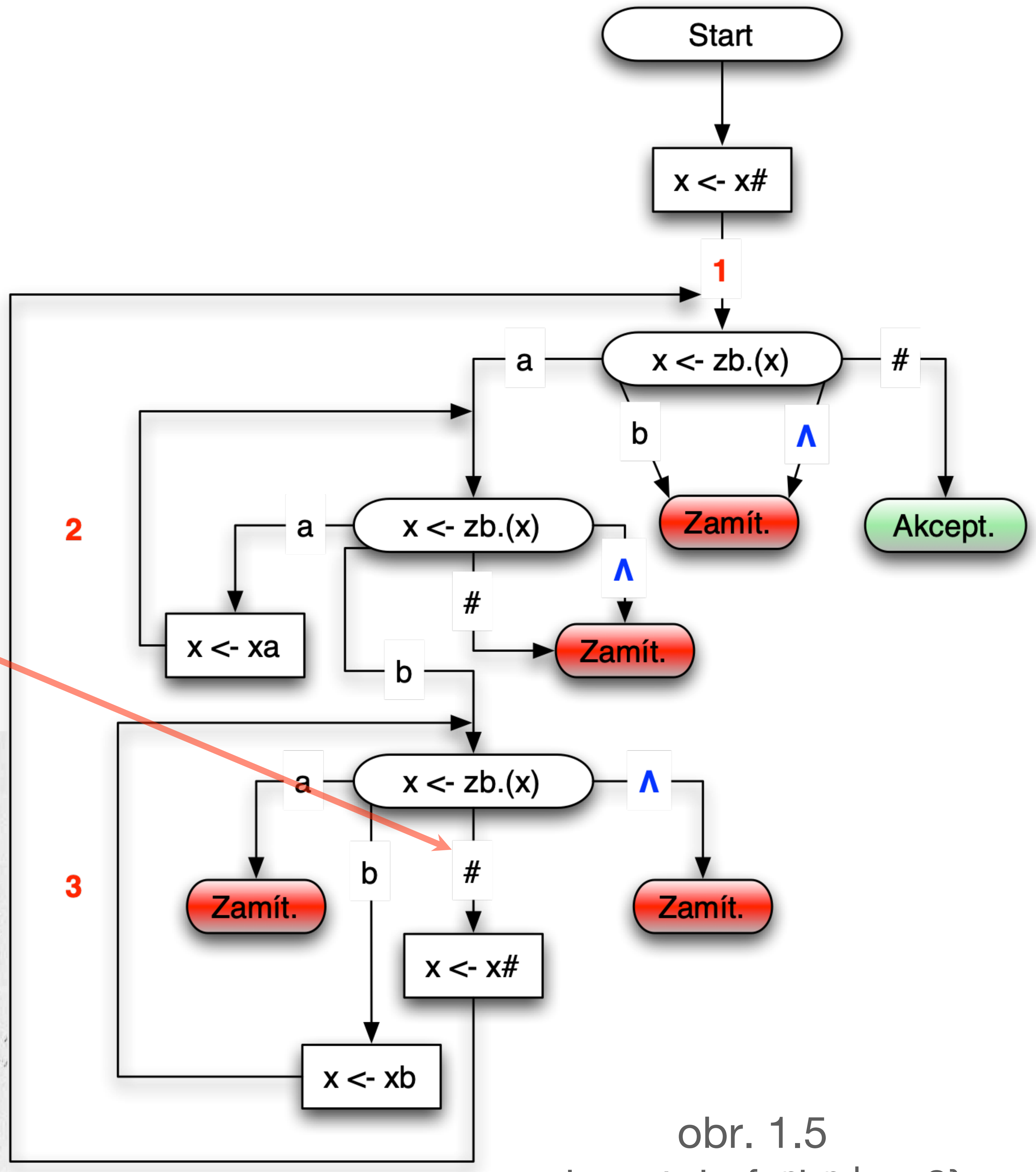


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

ab

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leq i \leq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

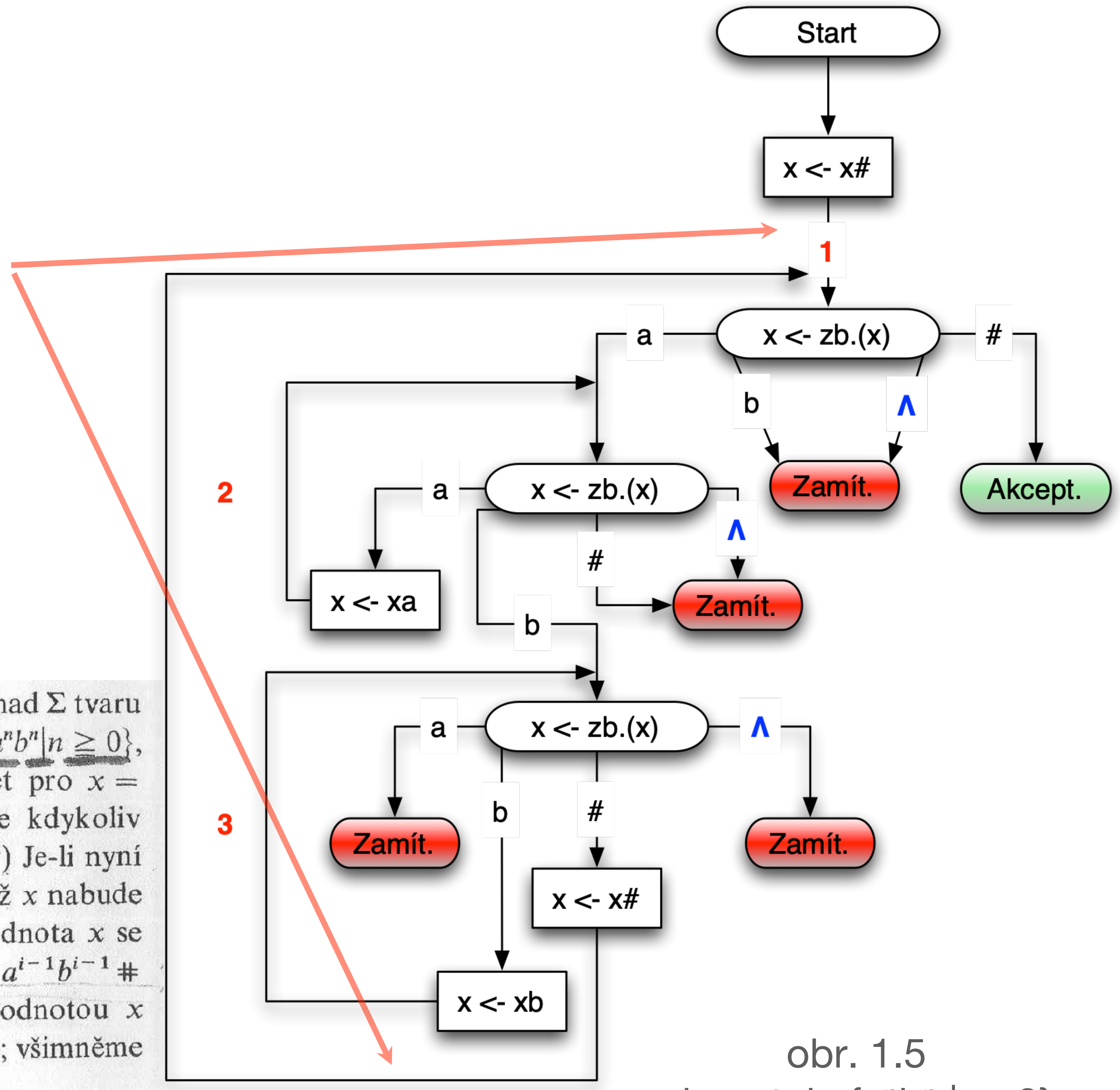


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

aabb# byl transformován na ab#

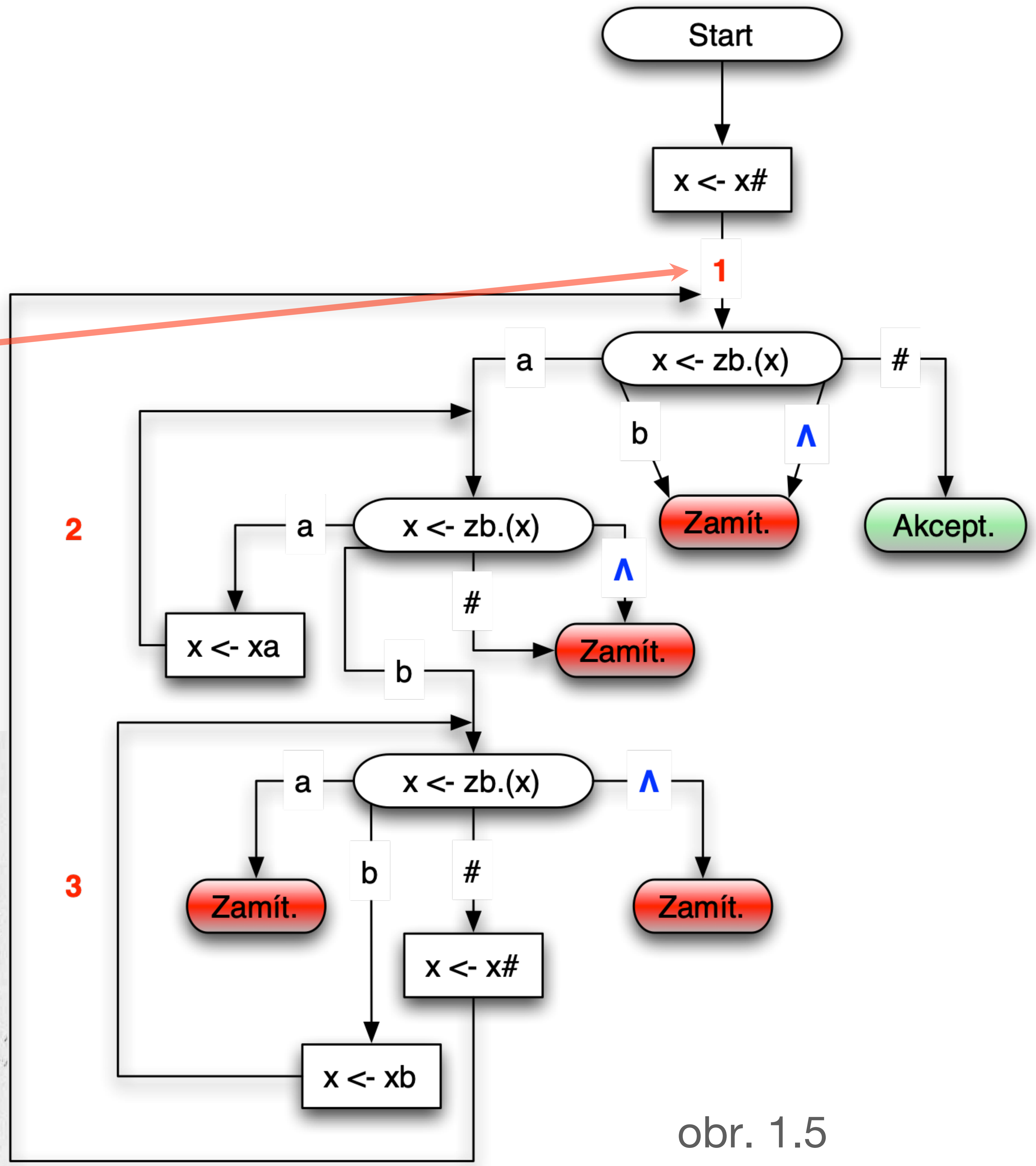
Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leq i \leq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol # na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol # a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■



obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

ab#



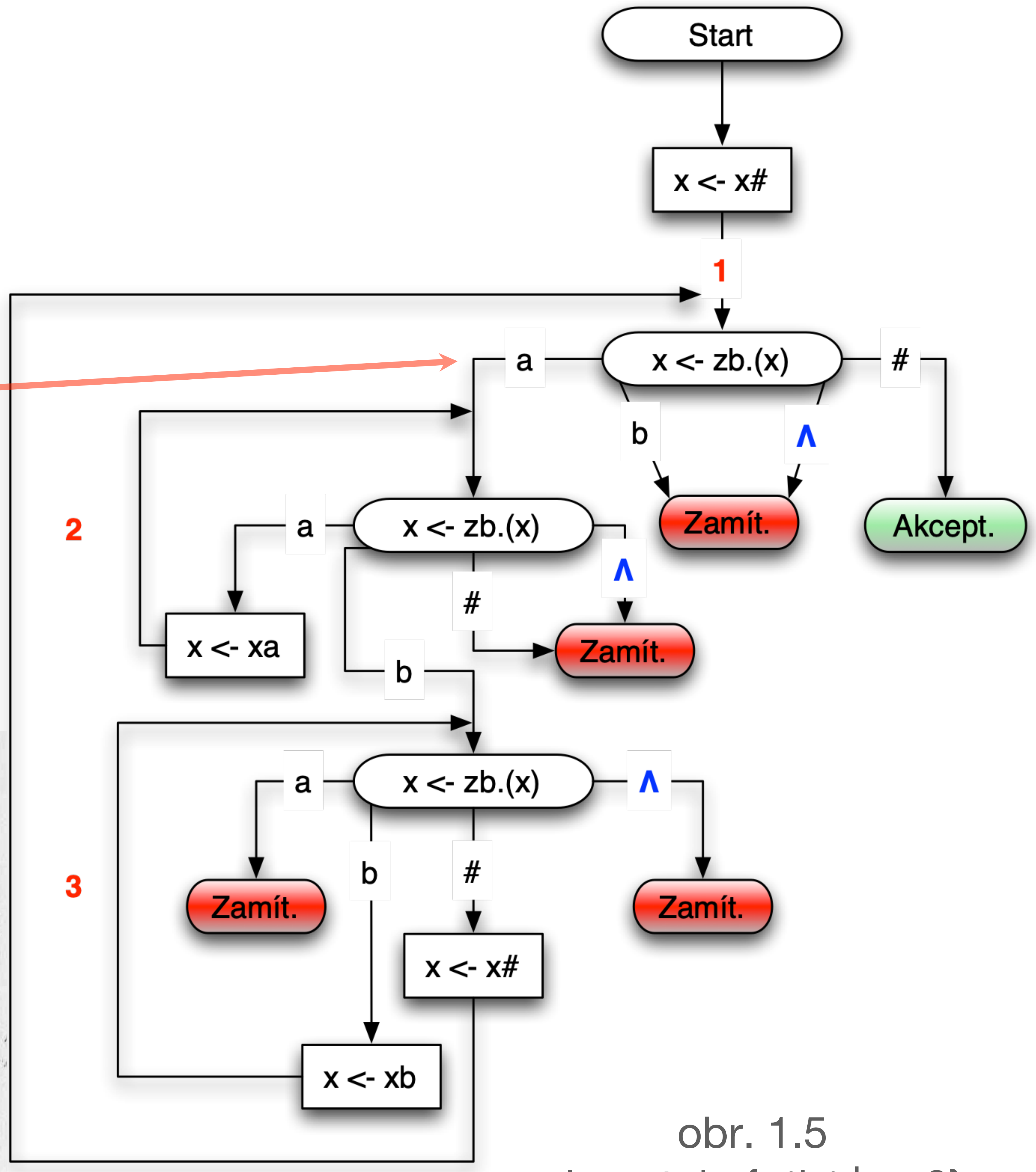
obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n \mid n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

PS příklad

b#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. $akceptuje(M_2) = \{a^n b^n | n \geqq 0\}$, $zamítá(M_2) = \Sigma^* - akceptuje(M_2)$, $cykluje(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

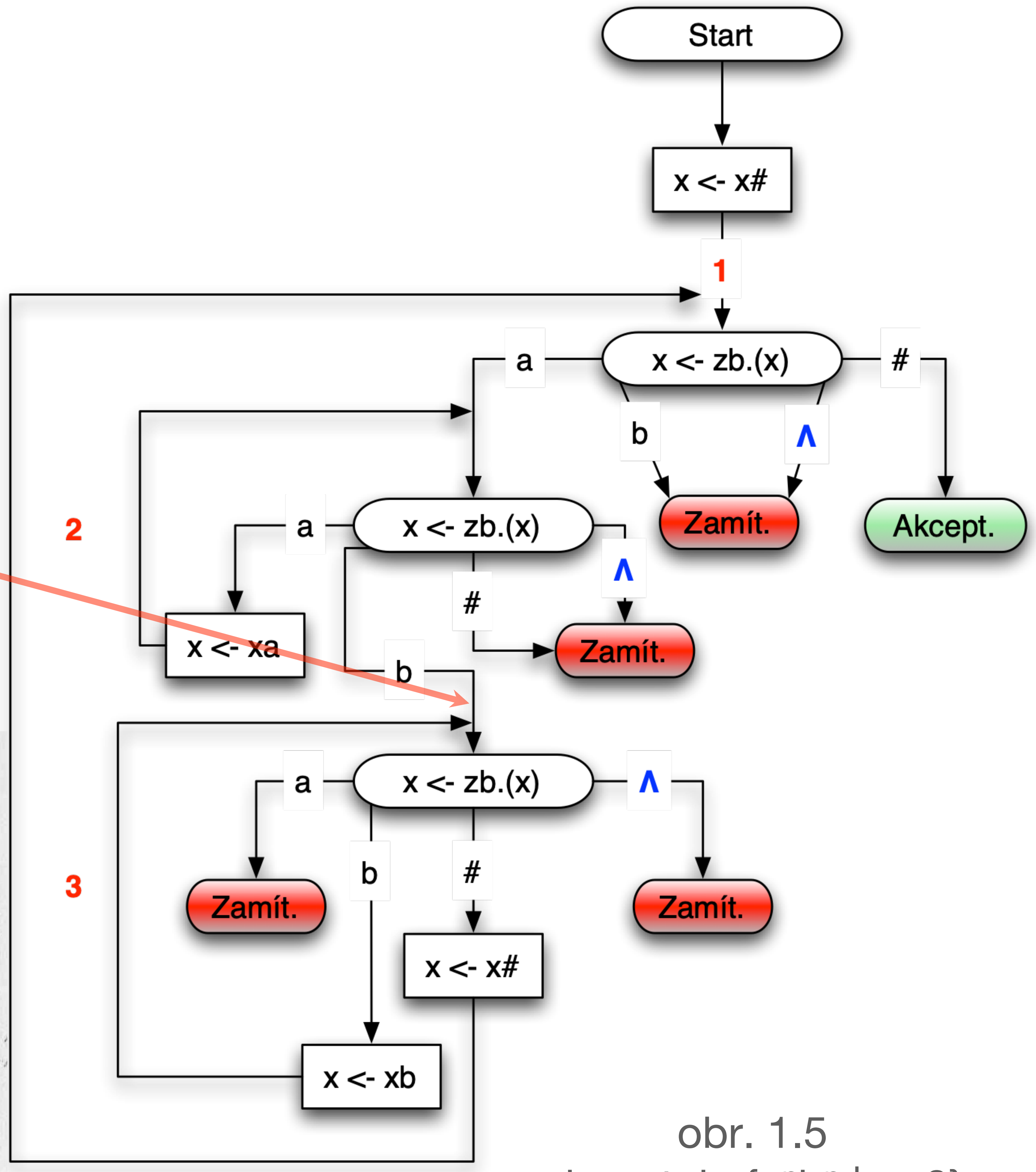


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

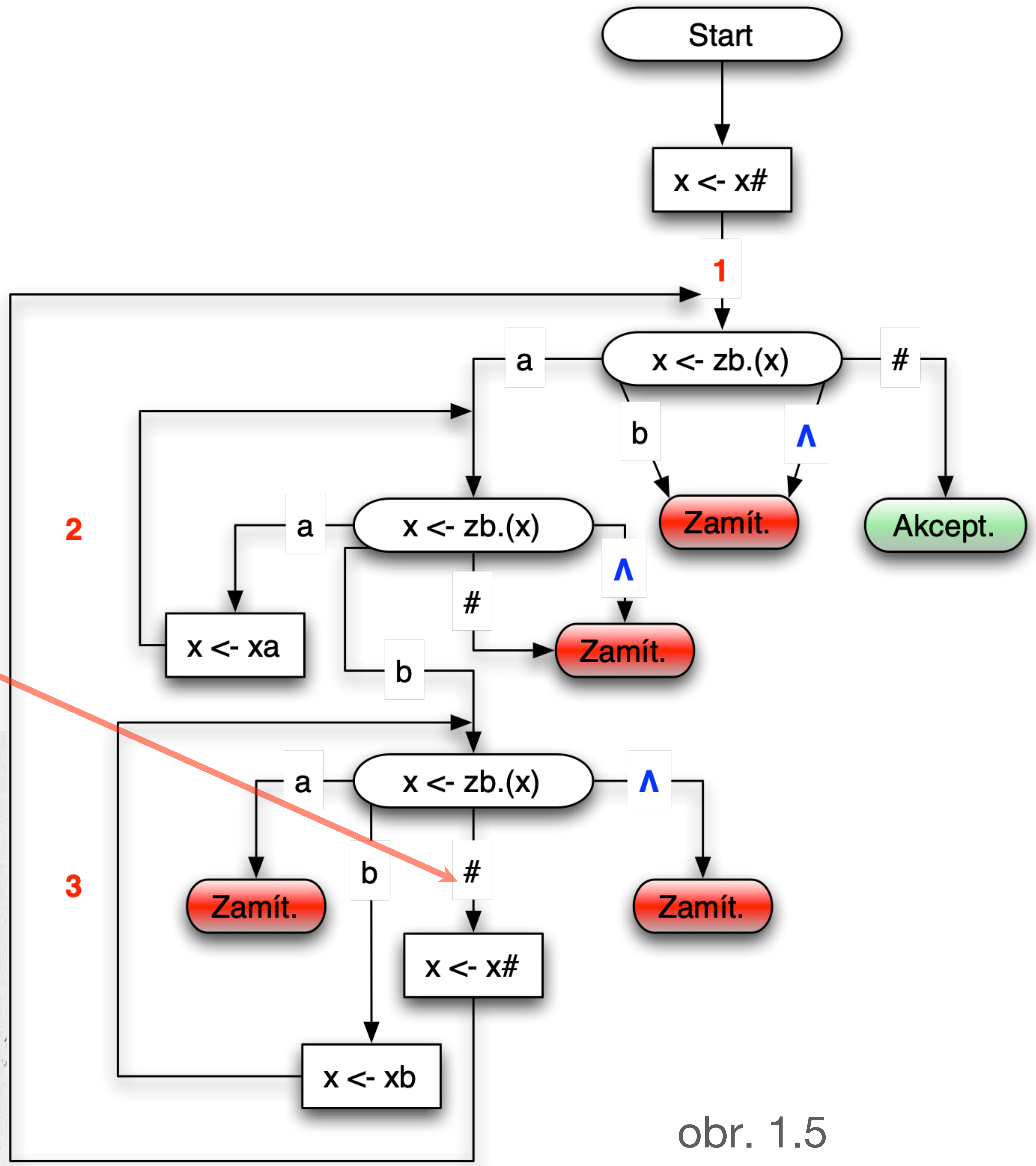


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

Λ

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geqq 0\}$, *zamítá*$(M_2) = \Sigma^* -$ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i - 1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i - 1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ■

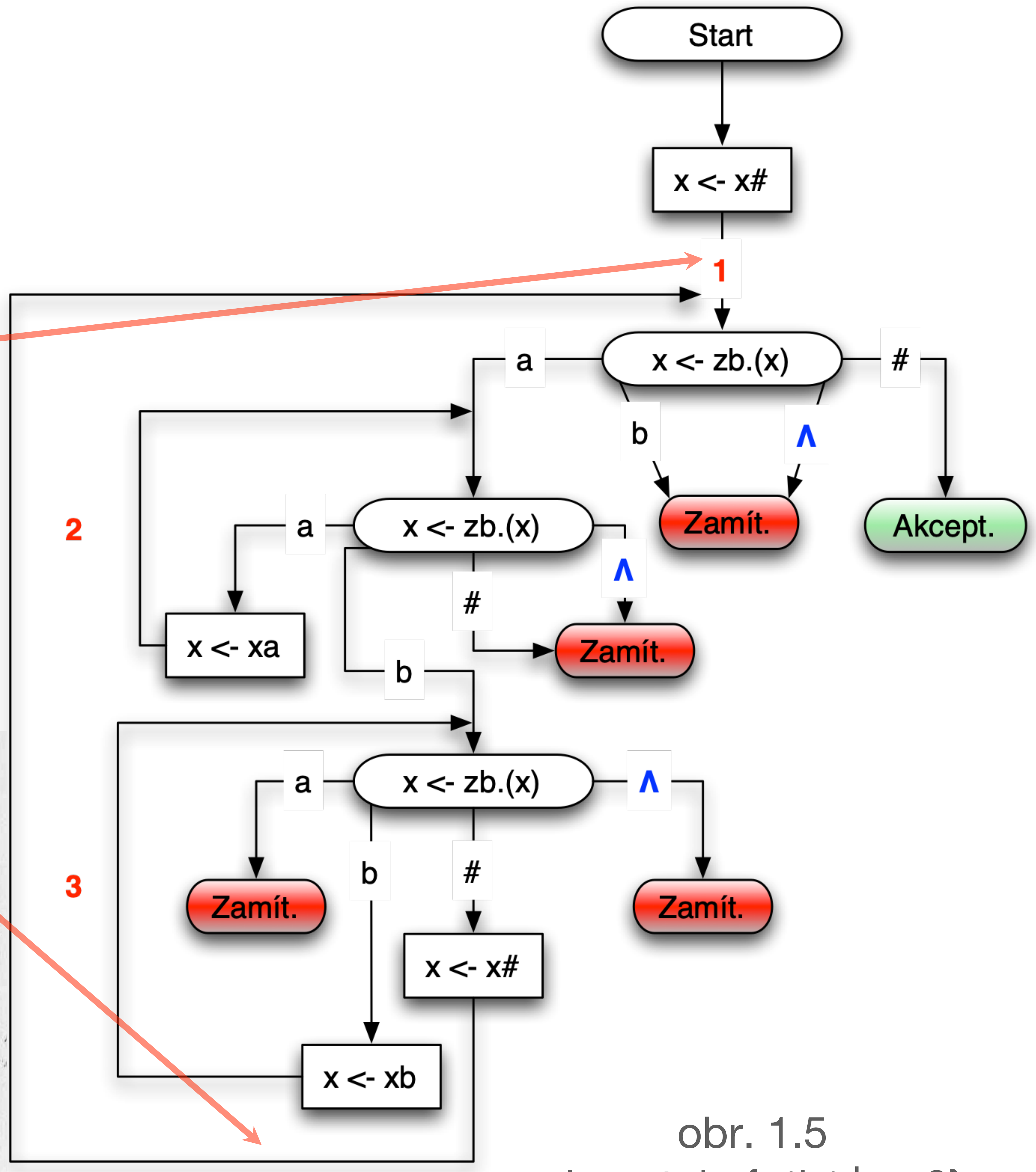


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad Σ, tj. $akceptuje(M_2) = \{a^n b^n | n \geqq 0\}$, $zamítá(M_2) = \Sigma^* - akceptuje(M_2)$, $cykluje(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geqq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leqq i \leqq n$.[2] Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol $\#$ na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol $\#$ a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎

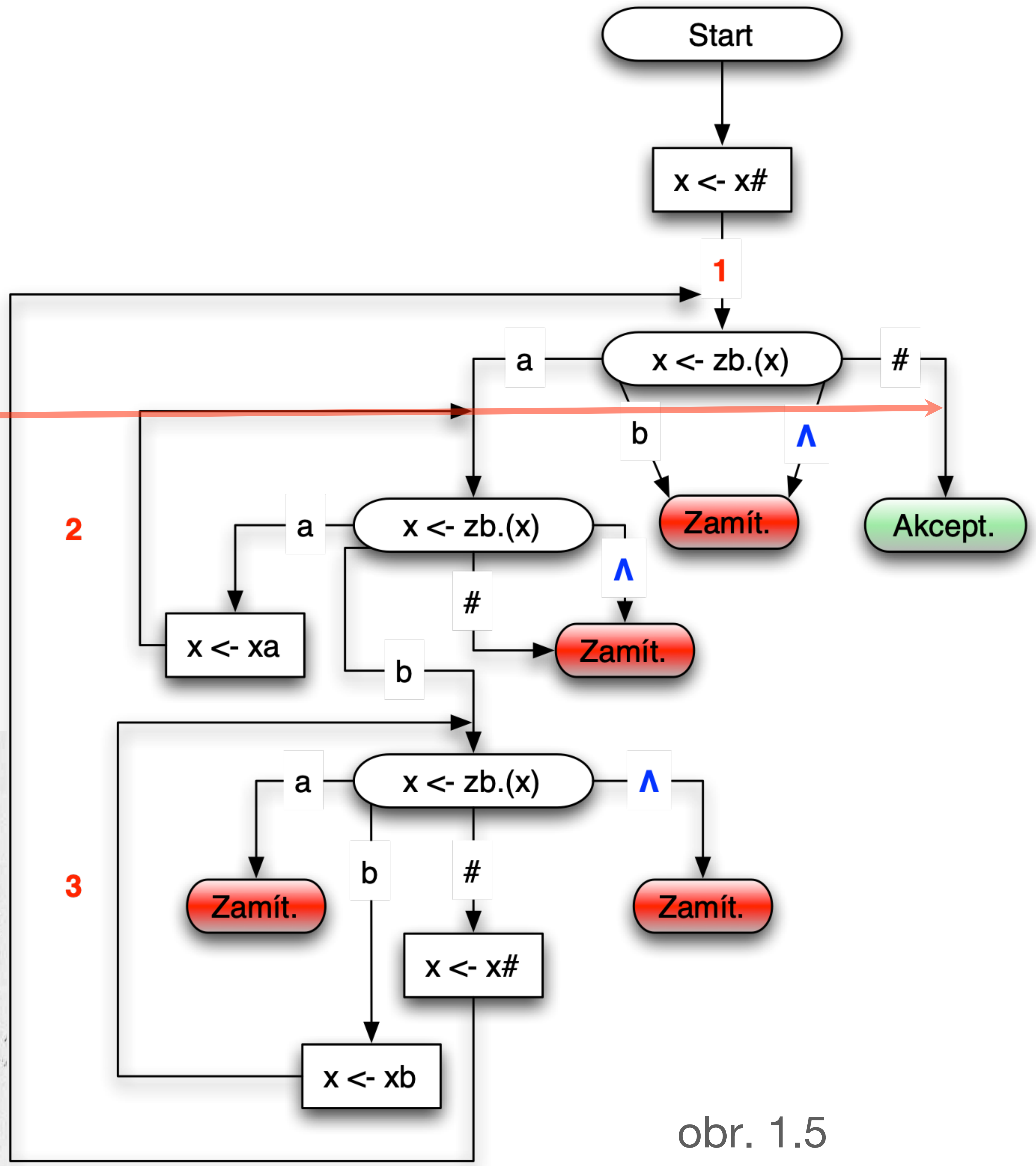


obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

PS příklad

#

Postův stroj M_2 nad $\Sigma = \{a, b\}$ z obr. 1.5 akceptuje všechna slova nad Σ tvaru $a^n b^n$, $n \geq 0$, a zamítá všechna ostatní slova nad Σ, tj. *akceptuje*$(M_2) = \{a^n b^n | n \geq 0\}$, *zamítá*$(M_2) = \Sigma^* - $ *akceptuje*(M_2), *cykluje*$(M_2) = \emptyset$. Sledujme výpočet pro $x = a^n b^n$ při nějakém pevném $n \geq 0$. Jádro důkazu spočívá v tom, že kdykoliv stroj prochází bodem 1, má x hodnotu $a^i b^i \#$ pro vhodné i, $0 \leq i \leq n$.[2]) Je-li nyní na začátku slova x písmeno a, prochází výpočet $(i-1)$krát cyklem 2, až x nabude hodnoty $b^i \# a^{i-1}$; potom jde výpočet $(i-1)$krát cyklem 3, takže hodnota x se změní na $\# a^{i-1} b^{i-1}$; nakonec se přesune symbol # na konec, $x = a^{i-1} b^{i-1} \#$ a celý proces se opakuje od bodu 1. Nakonec pro $i = 0$ bude hodnotou x pouze symbol # a výpočet přejde do závěrečného akceptujícího příkazu; všimněme si, že toto je jediný způsob jak dosáhnout příkazu AKCEPTOVÁNO. ∎



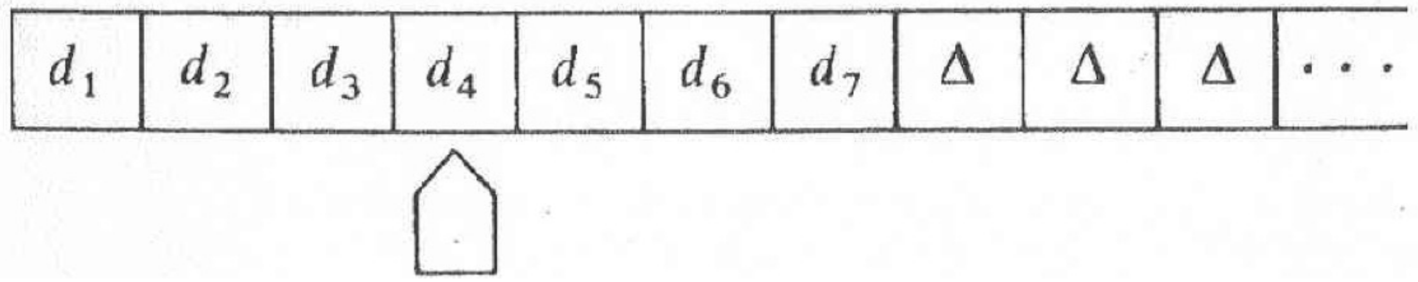
obr. 1.5
akceptuje $\{a^n b^n \mid n \geq 0\}$

Ekvivalence mezi PS a TS

Věta 1.3 (Post). *Třída Postových strojů nad abecedou Σ je stejně silná jako třída Turingových strojů nad Σ.* Jinak řečeno, ke každému Postovu stroji nad Σ existuje ekvivalentní Turingův stroj nad Σ a obráceně.

Důkaz. Ukážeme nejprve, že každý Turingův stroj nad $\Sigma = \{a, b\}$ může být simulován vhodným ekvivalentním Postovým strojem nad $\Sigma = \{a, b\}$. Hlavní idea této simulace spočívá v zápisu obsahu pásky a pozice hlavy v každém okamžiku výpočtu Turingova stroje pomocí vhodné hodnoty proměnné x Postova stroje.

Jsou-li např. v některém okamžiku výpočtu Turingova stroje zápis na pásce a poloha hlavy takové, jak ukazuje obrázek



kde každé $d_i \in \Pi$ a hlava čte symbol d_4, pak tuto konfiguraci vyjádříme v Postově stroji zápisem

$$x = d_4 d_5 d_6 d_7 \# d_1 d_2 d_3 .$$

Ignorujeme tedy nekonečný počet symbolů Δ a zbývající část zápisu na pásce, tj. $d_1 d_2 \dots d_7$, „pootočíme" takovým způsobem, že nejlevější symbol slova x odpovídá tomu, nad kterým je hlava Turingova stroje. Speciální symbol $\#$ odděluje obě části zapsaného řetězu; předpokládá se, že $\# \notin \Pi$. Buď tedy $x = d_4 d_5 d_6 d_7 \# d_1 d_2 d_3$ a nechť další krok Turingova stroje je (d_4, β, R); v tomto případě změníme hodnotu x v Postově stroji na

$$x = d_5 d_6 d_7 \# d_1 d_2 d_3 \beta .$$

Je-li $x = d_4 d_5 d_6 d_7 \# d_1 d_2 d_3$ a následující krok Turingova stroje je (d_4, β, L), pak je hodnota x změněna na

$$x = d_3 \beta d_5 d_6 d_7 \# d_1 d_2$$[1]).

Konečné stroje

- Konečný stroj se zásobníkem je vývojový diagram s jedinou proměnnou x a zásobníkem z. Pracuje nad abecedou Σ.
- Vývojový diagram může obsahovat následující bloky: počáteční blok „START”, koncové bloky „AKCEPT” a „ZAMÍT”
- bloky příkazu přiřazení – příkaz přiřazení přidává symbol na začátek zá- sobníku z (přidává zleva)
- blok logického příkazu pro proměnnou x – logický příkaz umazává jeden symbol v proměnné x zleva a následně podle hodnoty tohoto symbolu větví program do několika větví
- blok logického příkazu pro zásobník z – logický příkaz umazává jeden sym- bol ze zásobníku z zleva a následně podle hodnoty tohoto symbolu větví program do několika větví

Konečné stroje - činnost

- Na počátku činnosti stroje je slovo w zápsáno do proměnné x, program začíná blokem „START”. Slovo w je akceptováno, jestliže se stroj v konečném počtu kroků dostane do stavu „AKCEPT”. Slovo w je zamítnuto jestriže se stroj v konečném počtu kroku dostane do stavu „ZAMÍT”. V ostatních případech stroj cykluje.

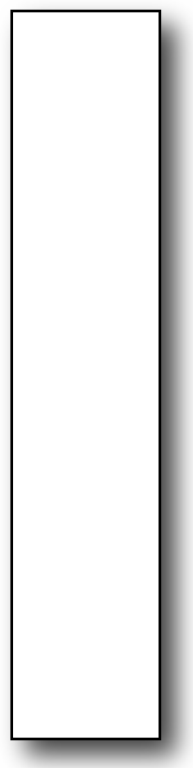
- Obdobně pracují i stroje s více zásobníky.

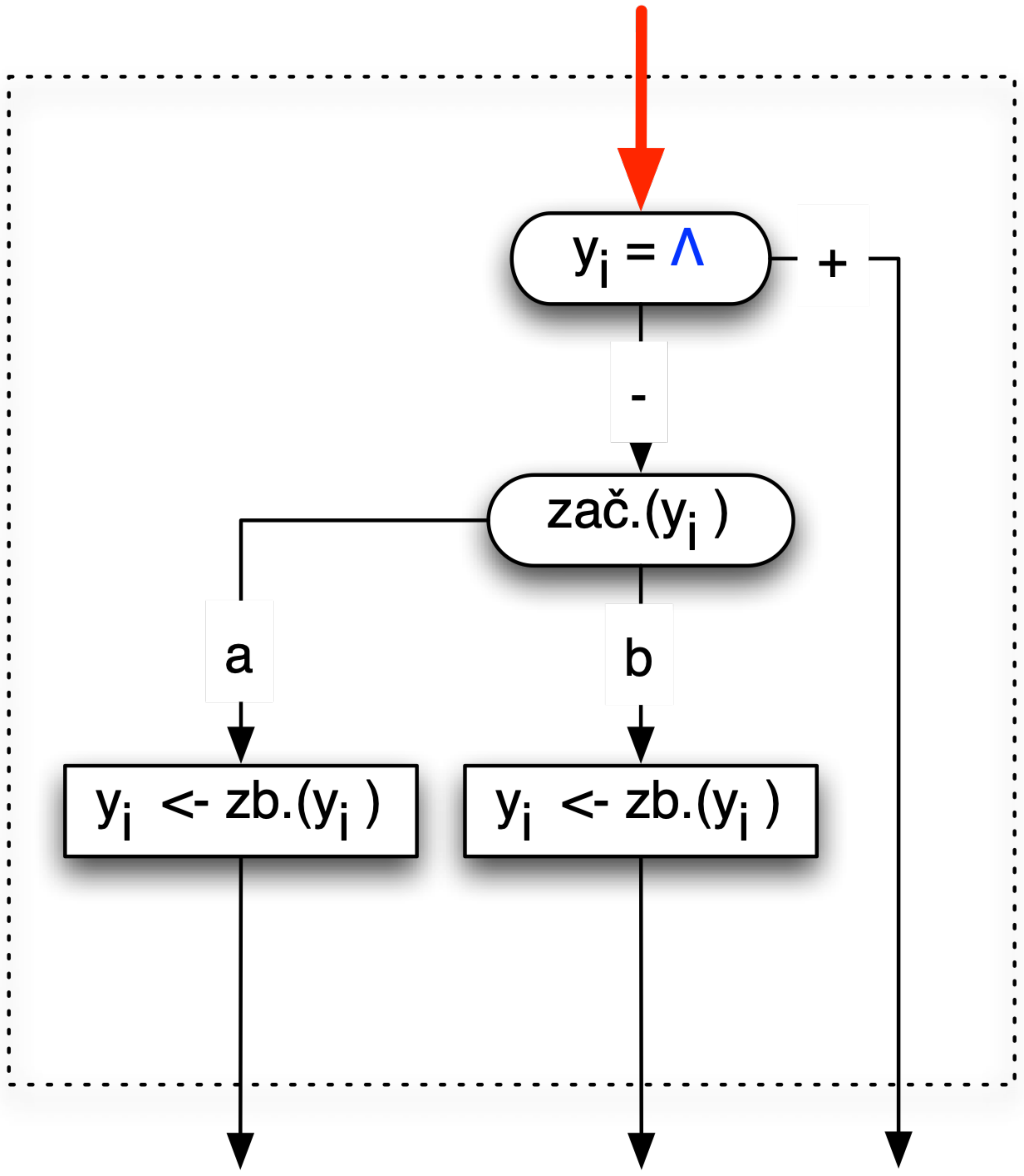
- Značná podobnost s Postovými sroji! Ale - U Postových strojů můžeme jednak číst a umazávat symboly z proměnné x zleva a jedna přidávat nové symboly zprava (paměťový typ FRONTA - FIFO). Postupnými modifikacemi obou konců tedy informace může “cirkulovat”. U konečných strojů se zásobníky nelze do paměťové proměnné “připisovat” zprava a zásobník je dán formou LIFO.

Konečné stroje - činnost

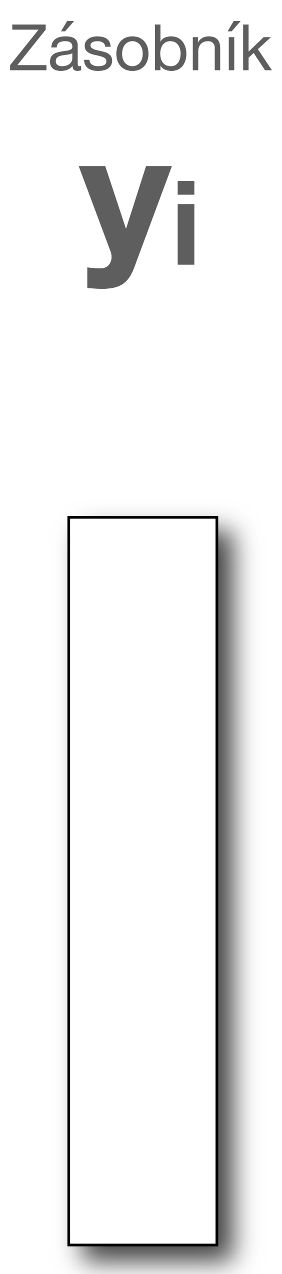
Zásobník

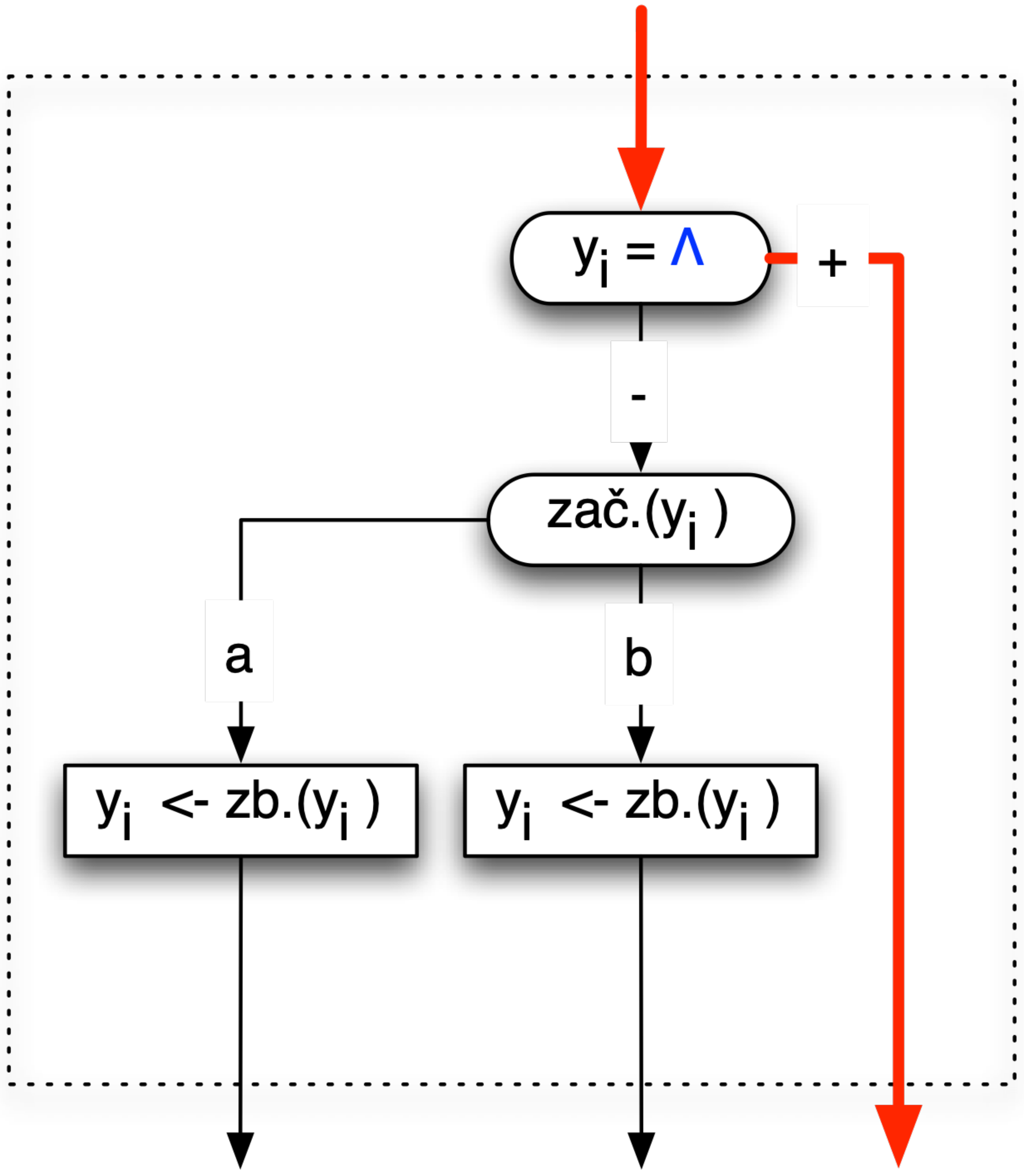
y_i





Konečné stroje - činnost

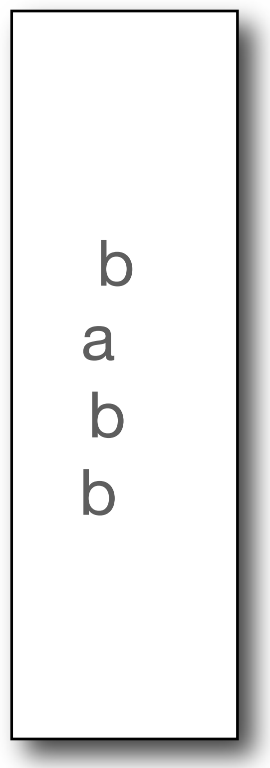


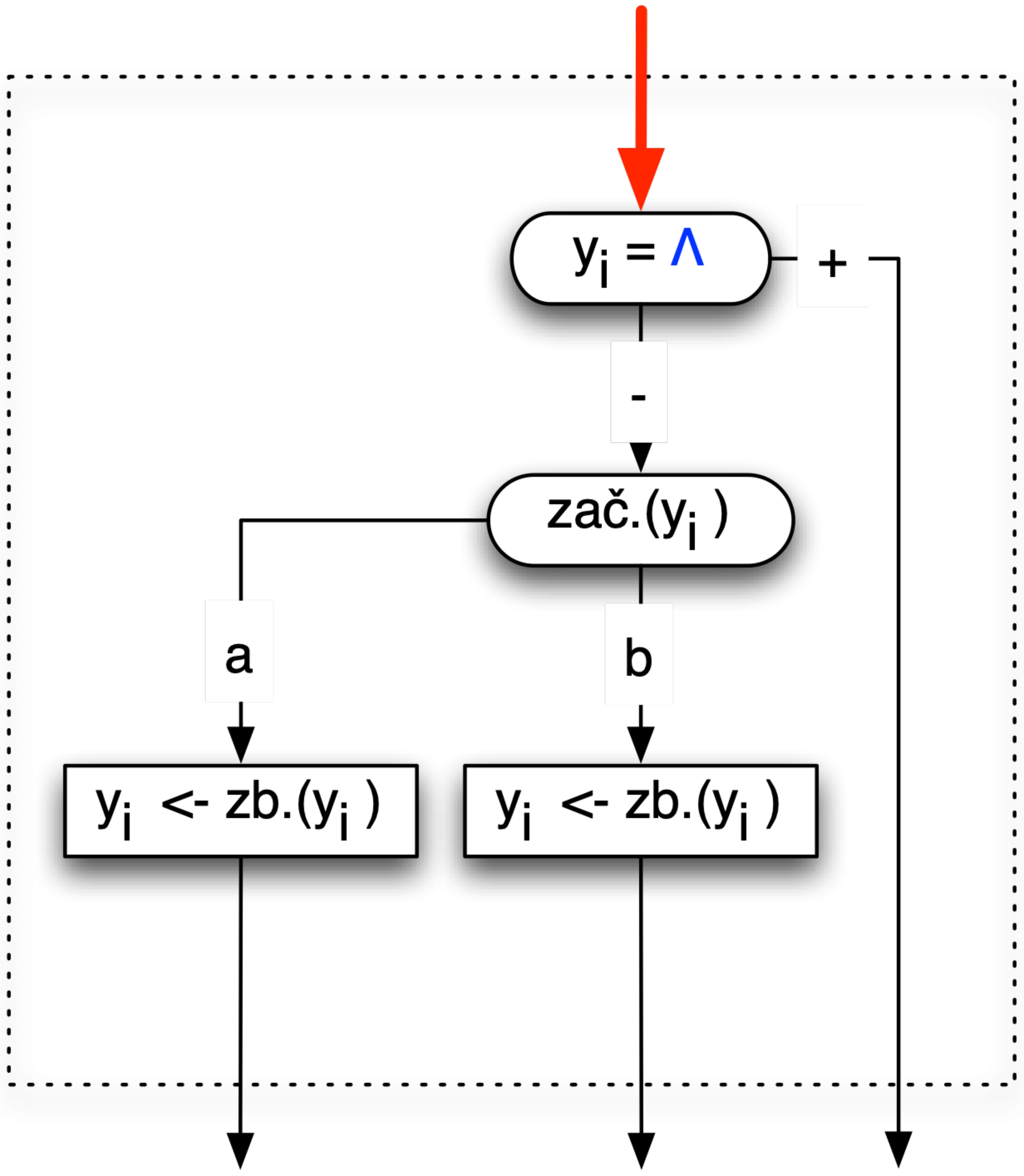


Konečné stroje - činnost

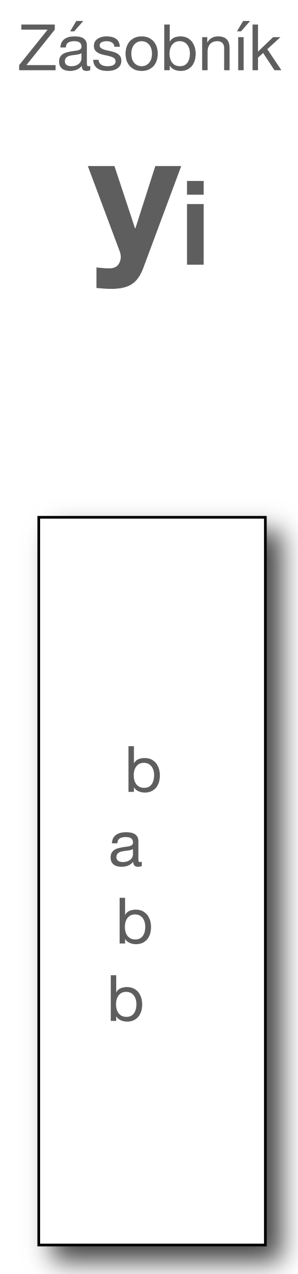
Zásobník

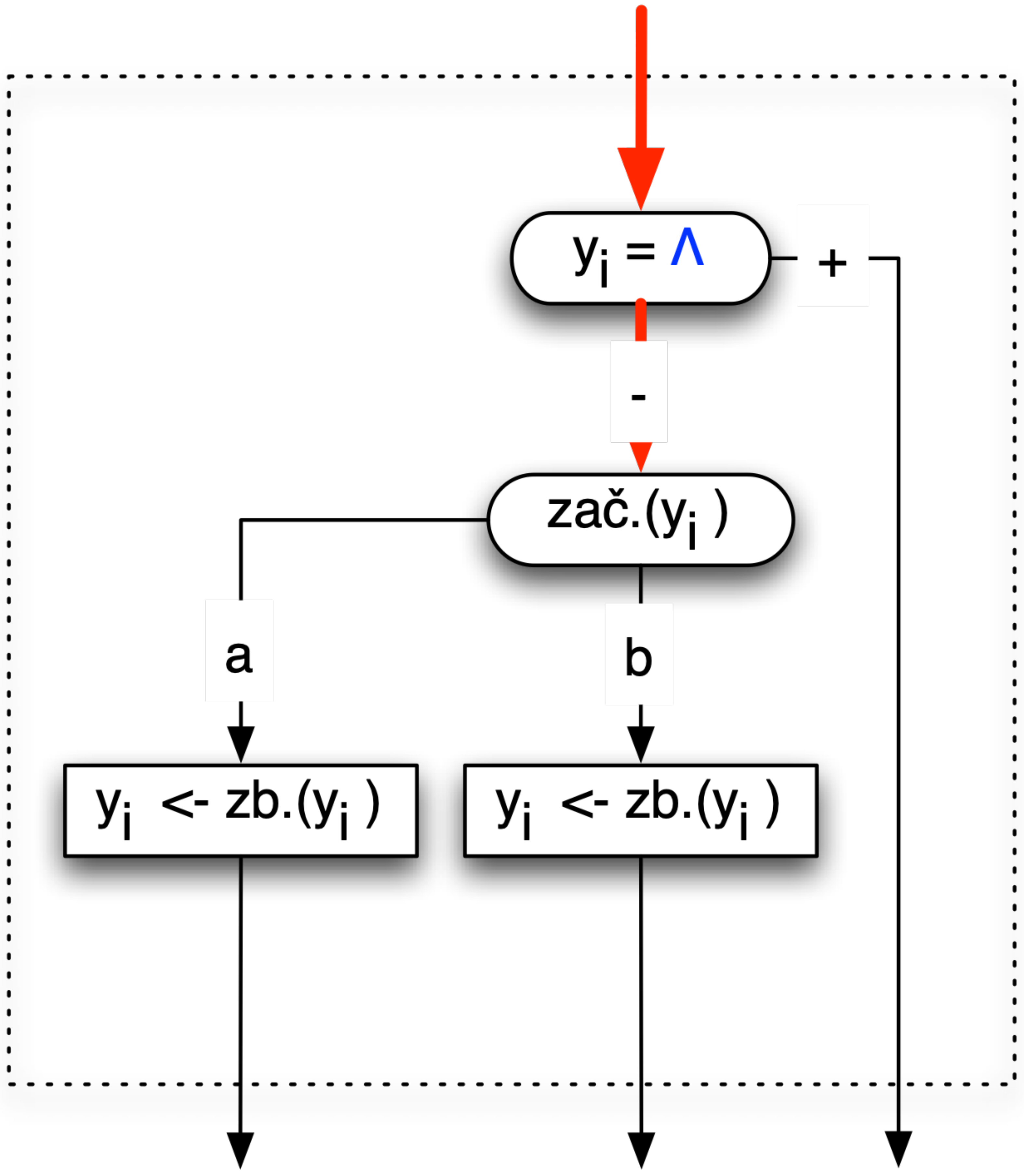
y_i



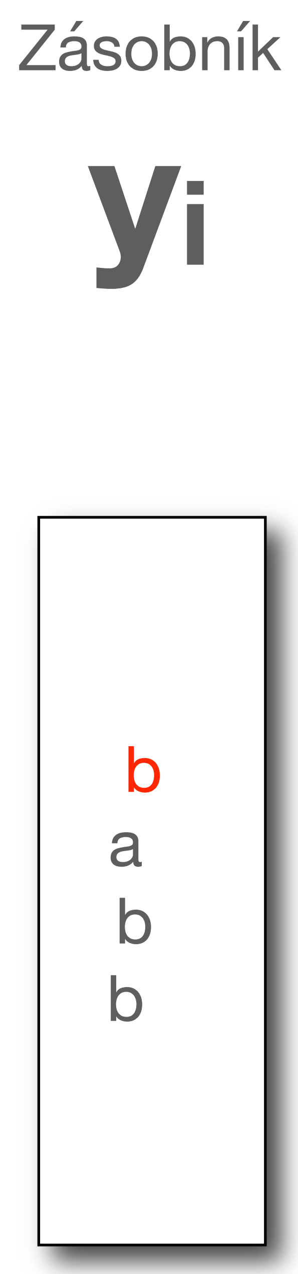


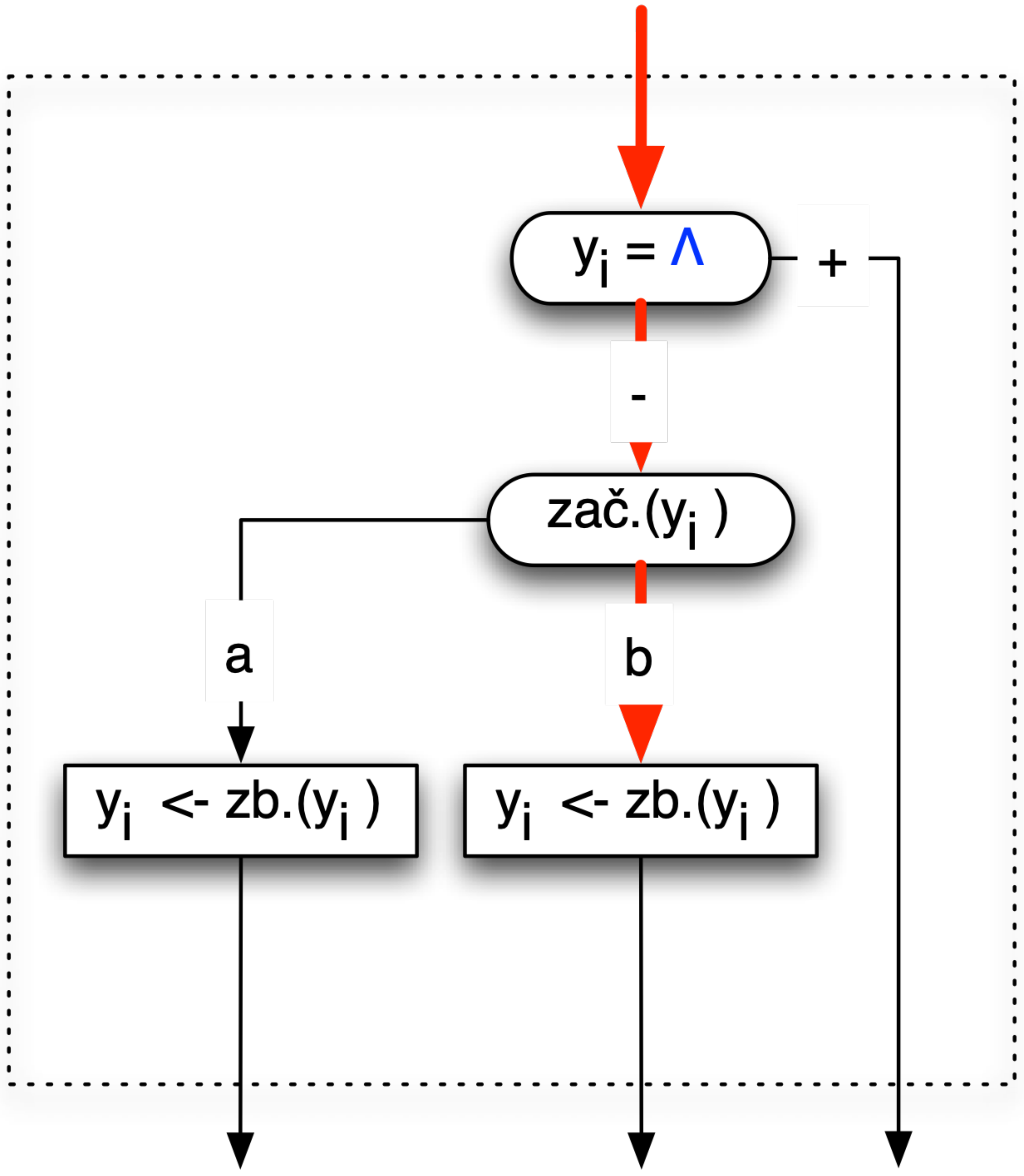
Konečné stroje - činnost



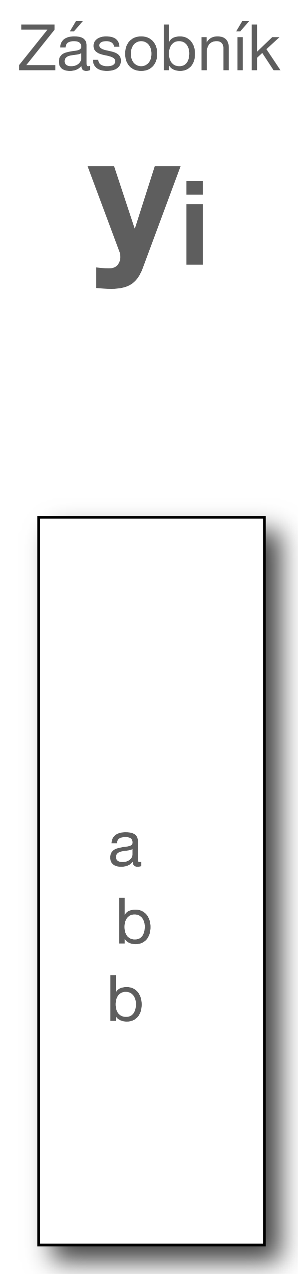


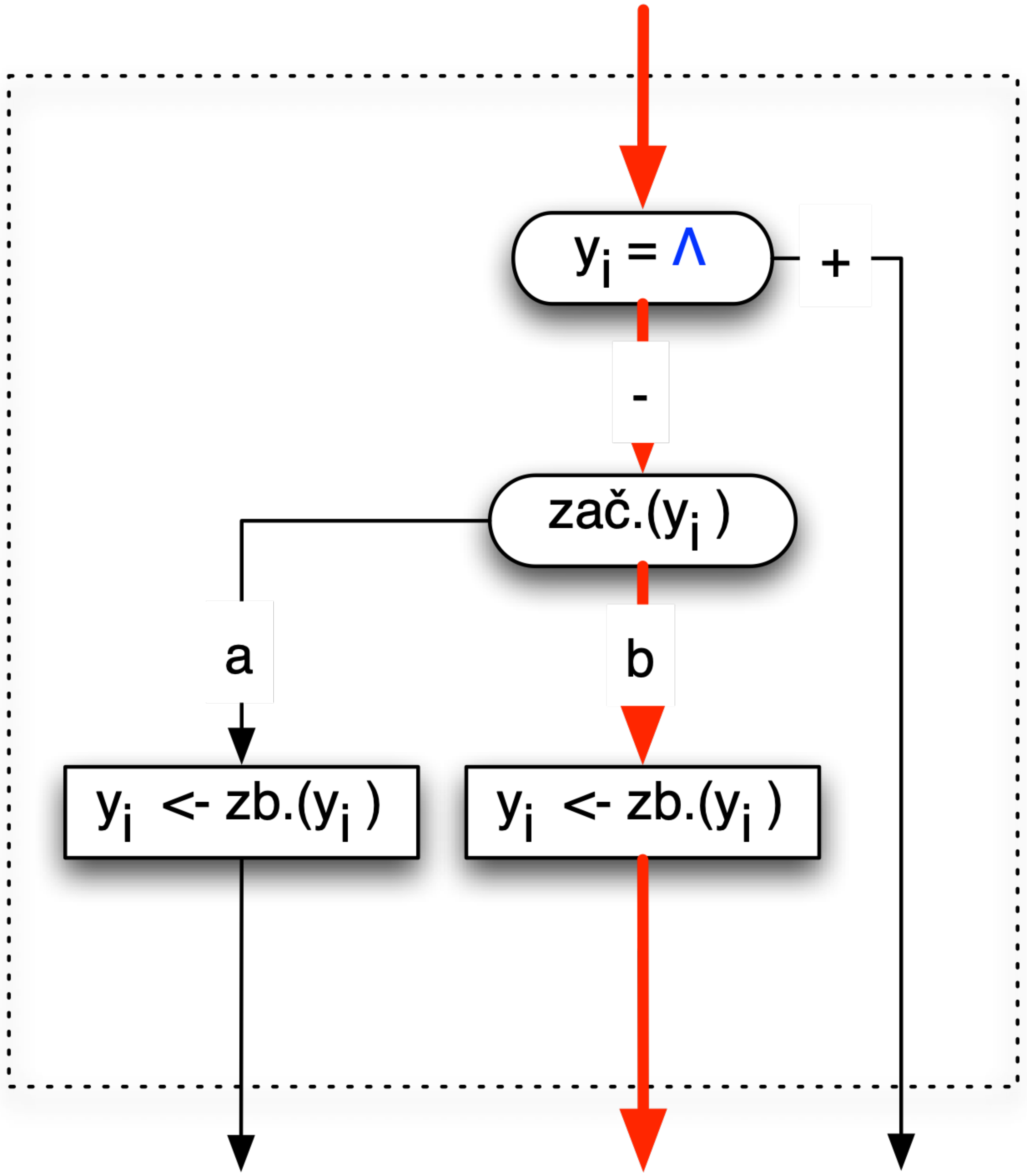
Konečné stroje - činnost





Konečné stroje - činnost



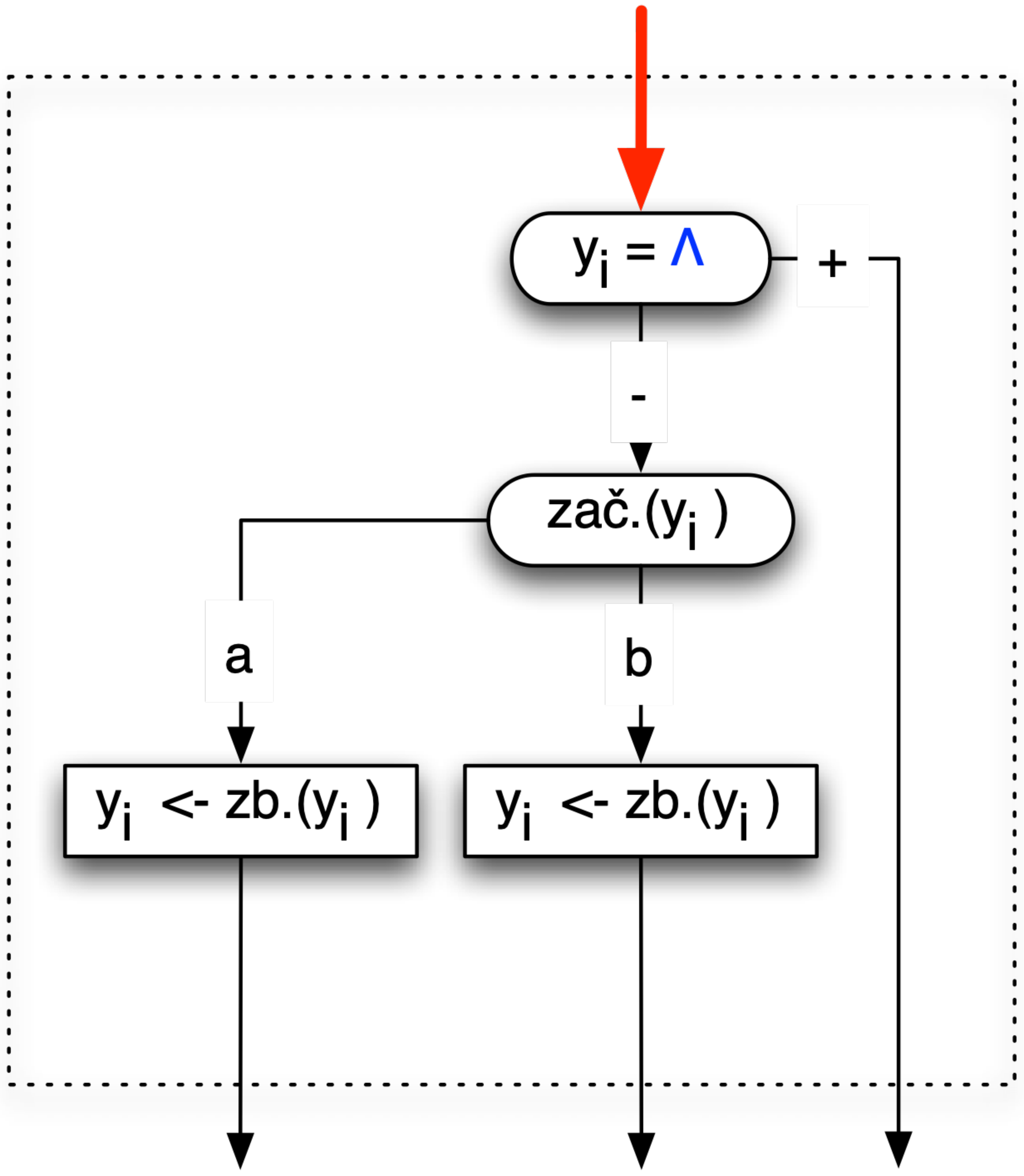


Konečné stroje - činnost

Zásobník

y_i

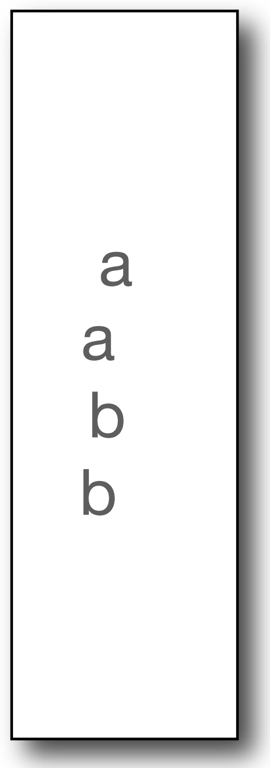
a
a
b
b

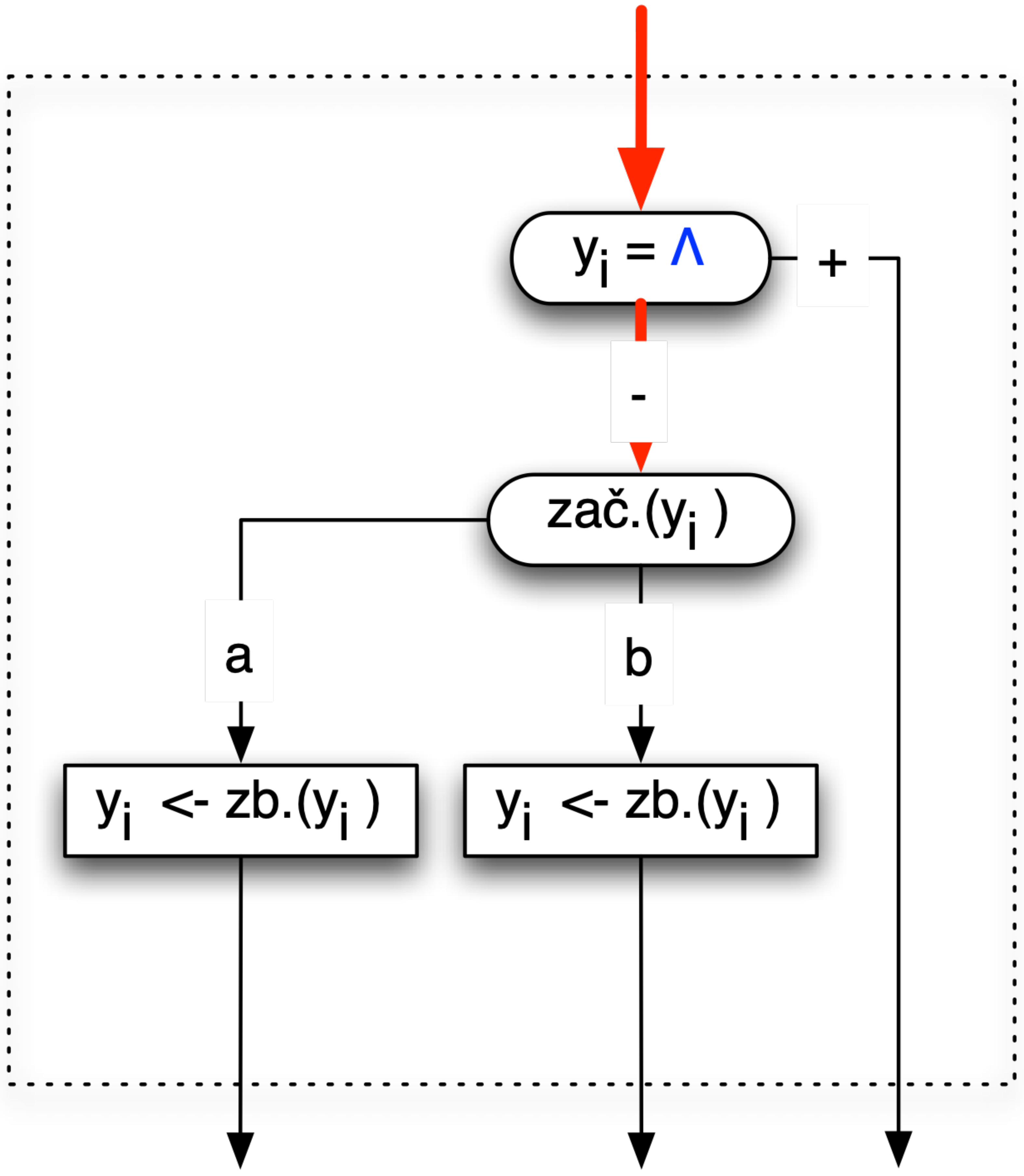


Konečné stroje - činnost

Zásobník

y_i

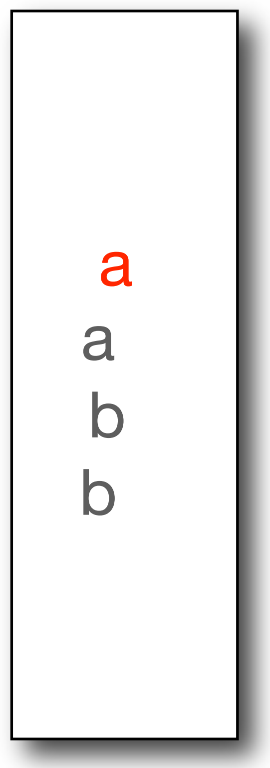


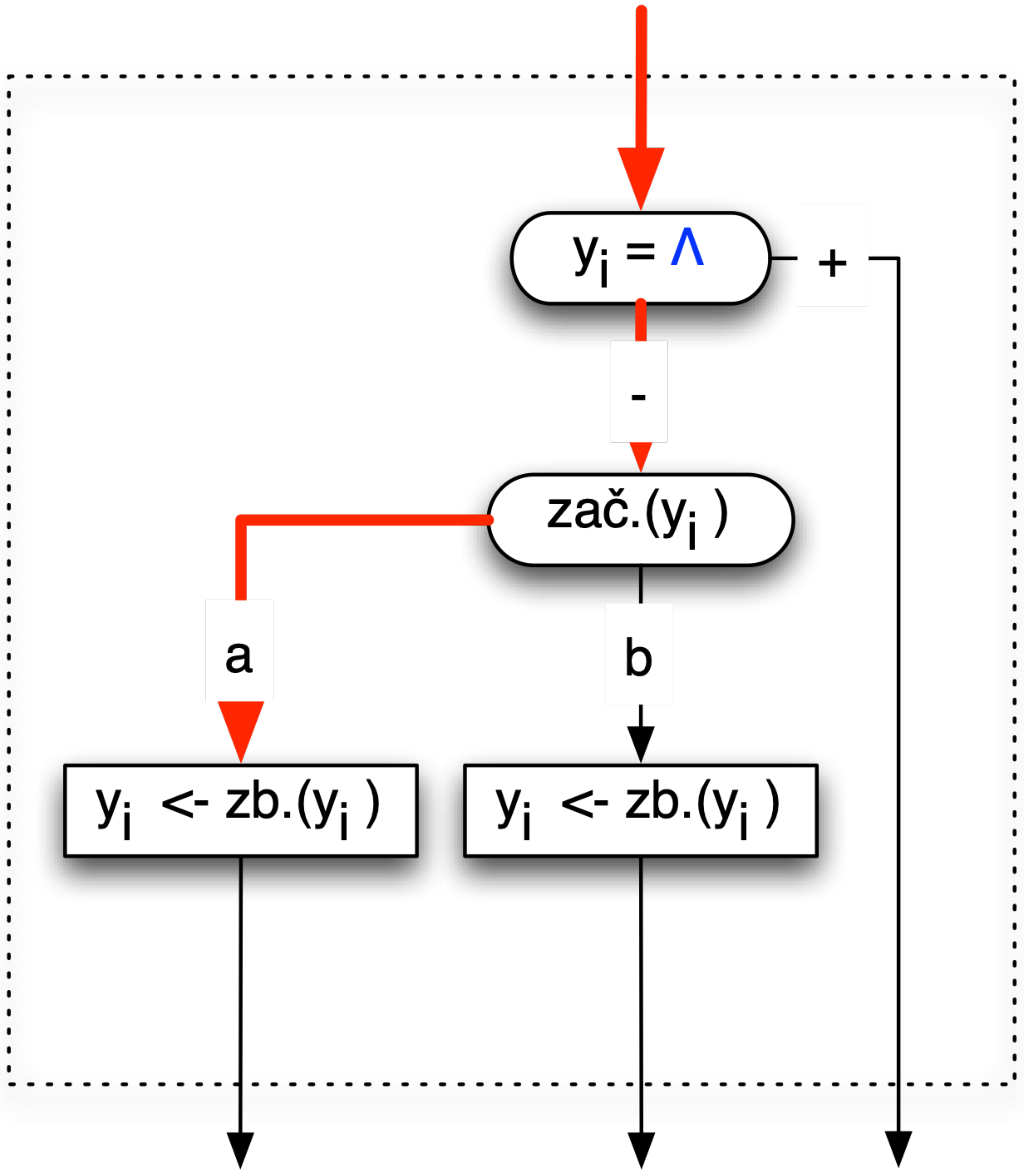


Konečné stroje - činnost

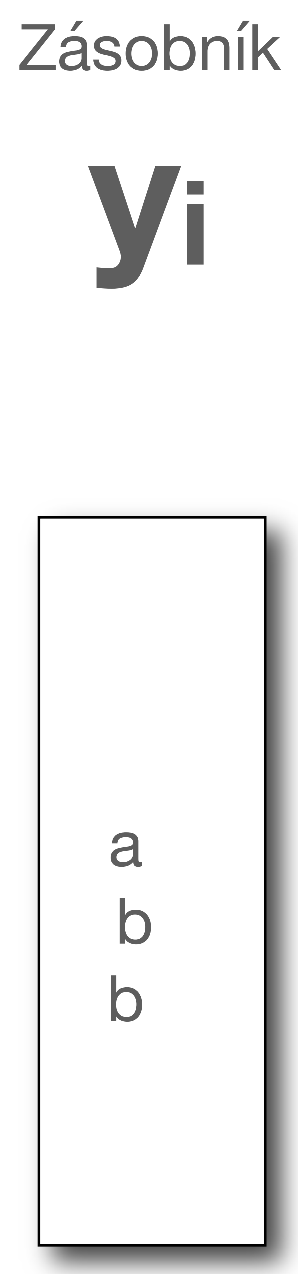
Zásobník

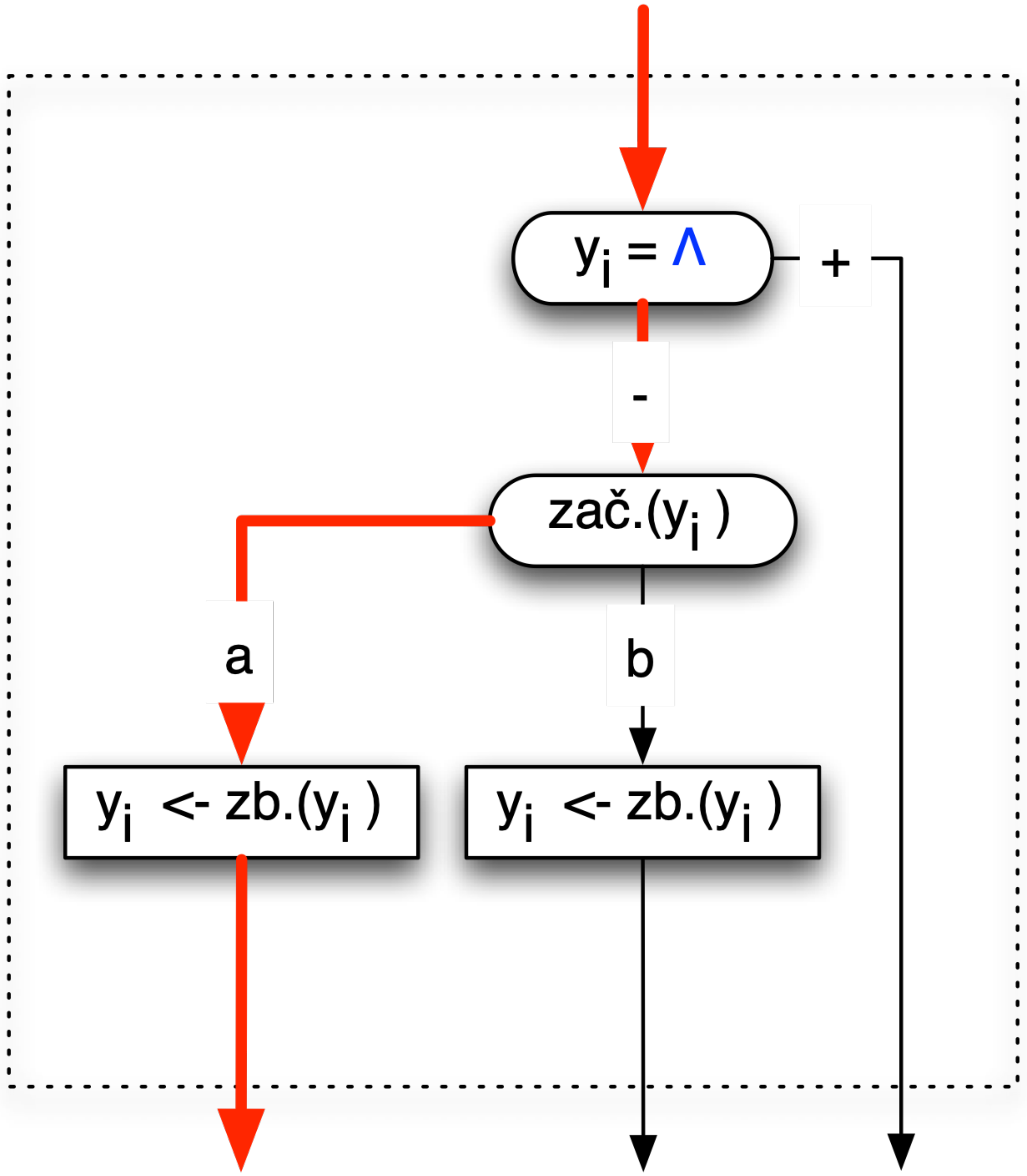
y_i





Konečné stroje - činnost

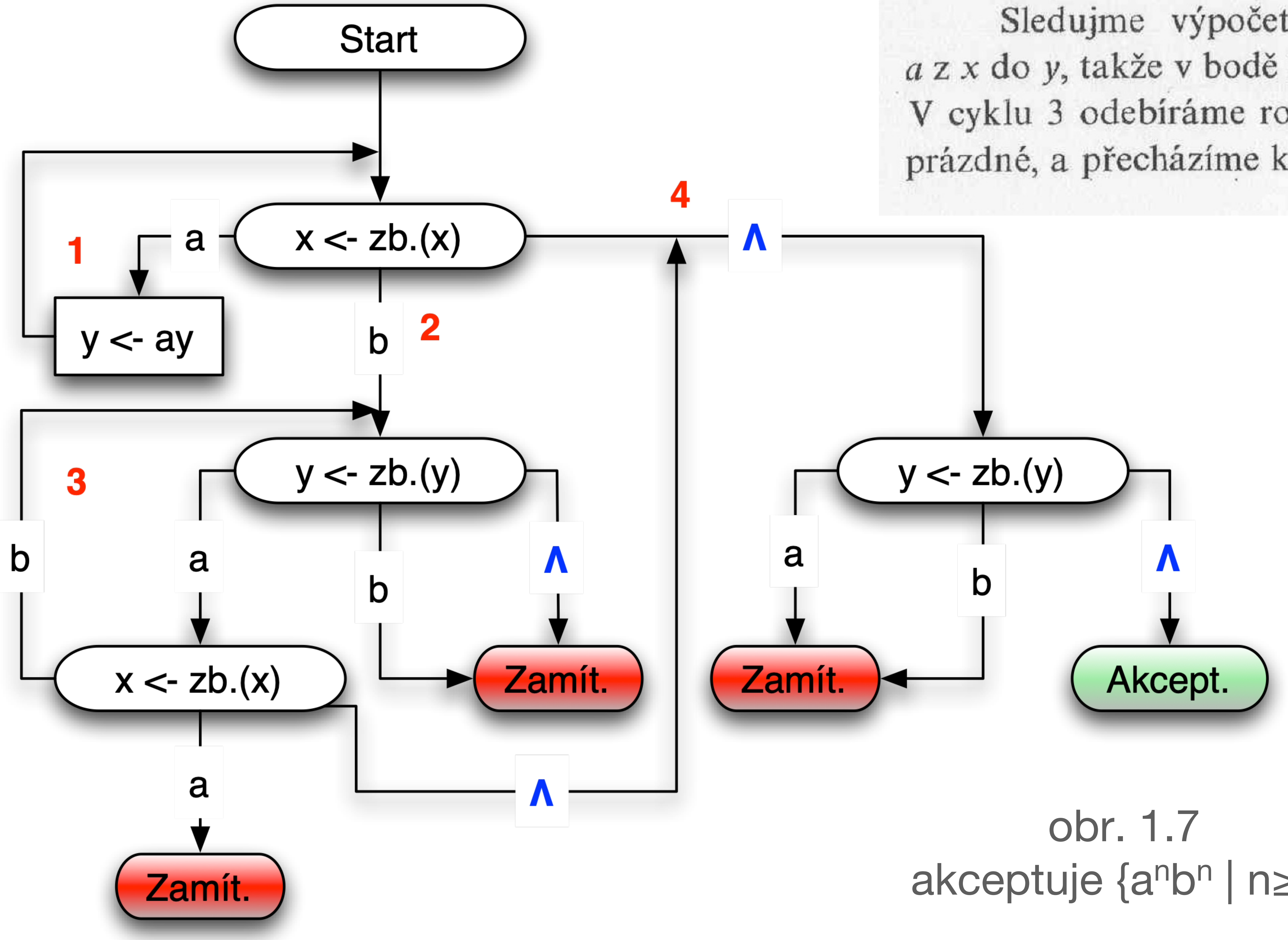




Konečné stroje - činnost

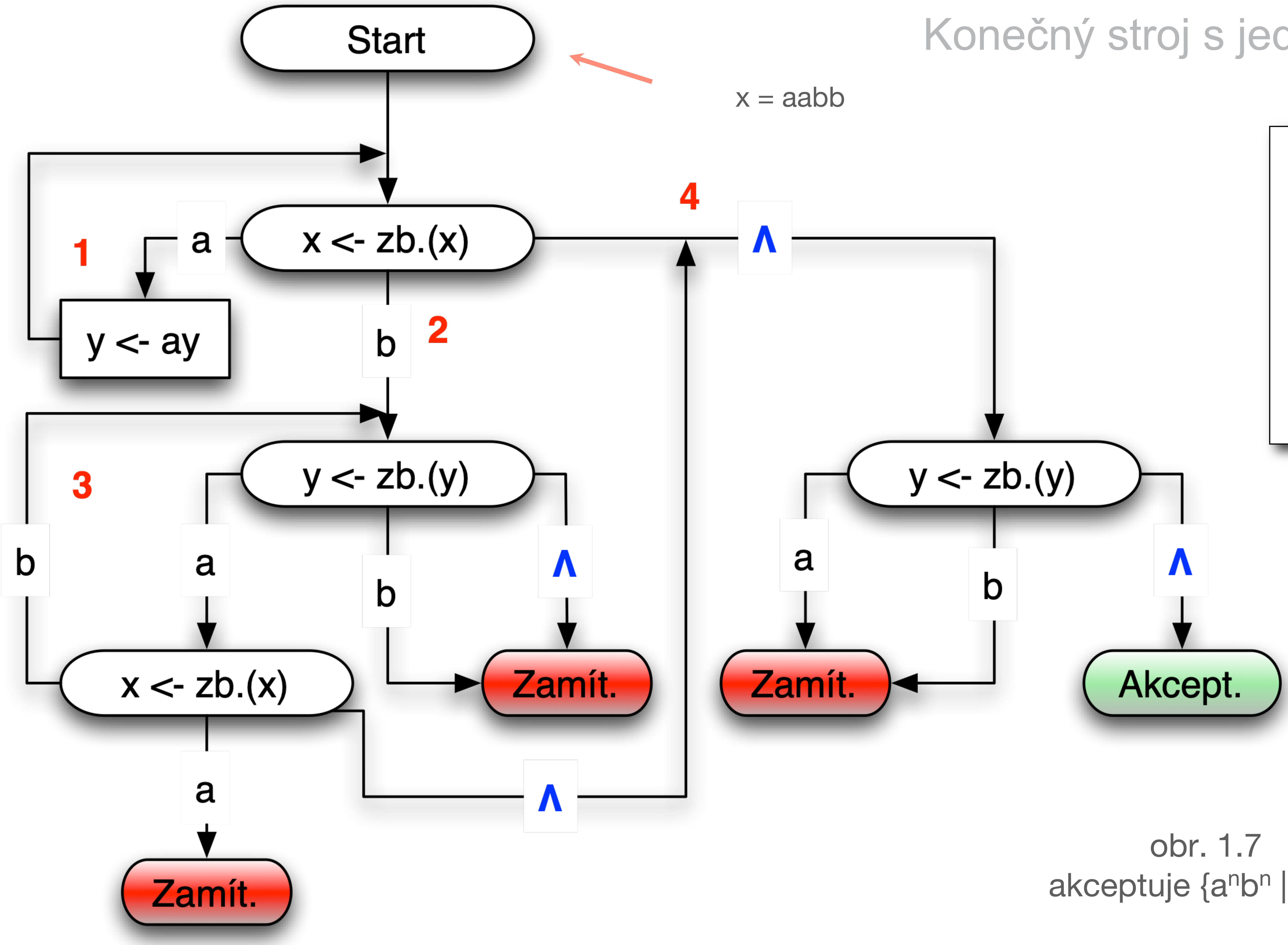
Konečný stroj M_4 s jedním zásobníkem y (viz obr. 1.7) akceptuje každé slovo tvaru $a^n b^n$, $n \geqq 0$, a zamítá všechna ostatní slova nad $\Sigma = \{a, b\}$. Je tedy *akceptuje*$(M_4) = \{a^n b^n | n \geqq 0\}$ a *zamítá*$(M_4) = \Sigma^* -$ *akceptuje*(M_4).

Sledujme výpočet pro $x = a^n b^n$. Nejprve se přesunou všechna písmena a z x do y, takže v bodě 2 výpočtu je hodnota y rovna a^n, zatímco v x zůstává b^{n-1}. V cyklu 3 odebíráme rovným dílem písmena a z y a písmena b z x, až x i y jsou prázdné, a přecházíme k bodu 4 výpočtu.



obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem

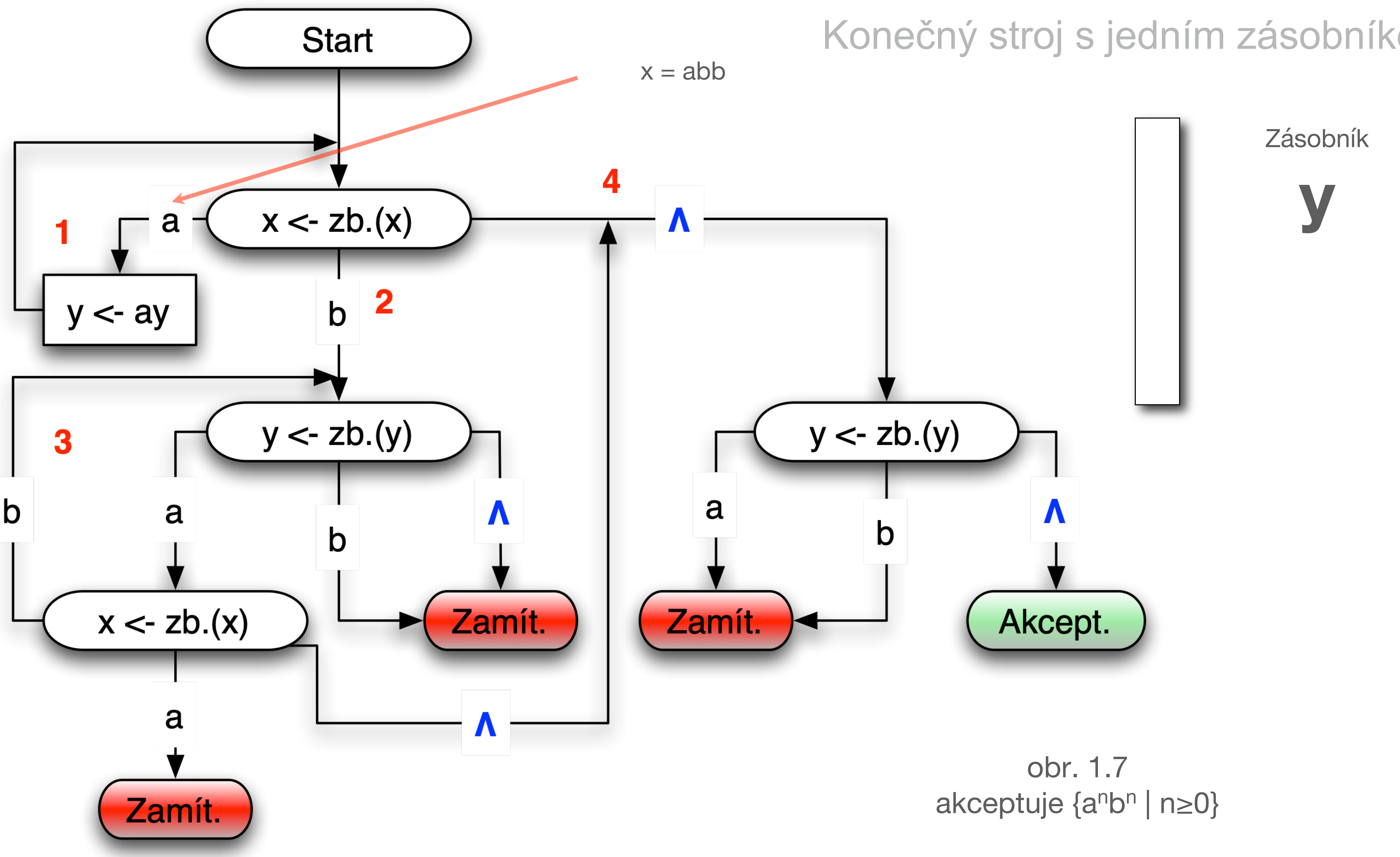


Zásobník

y

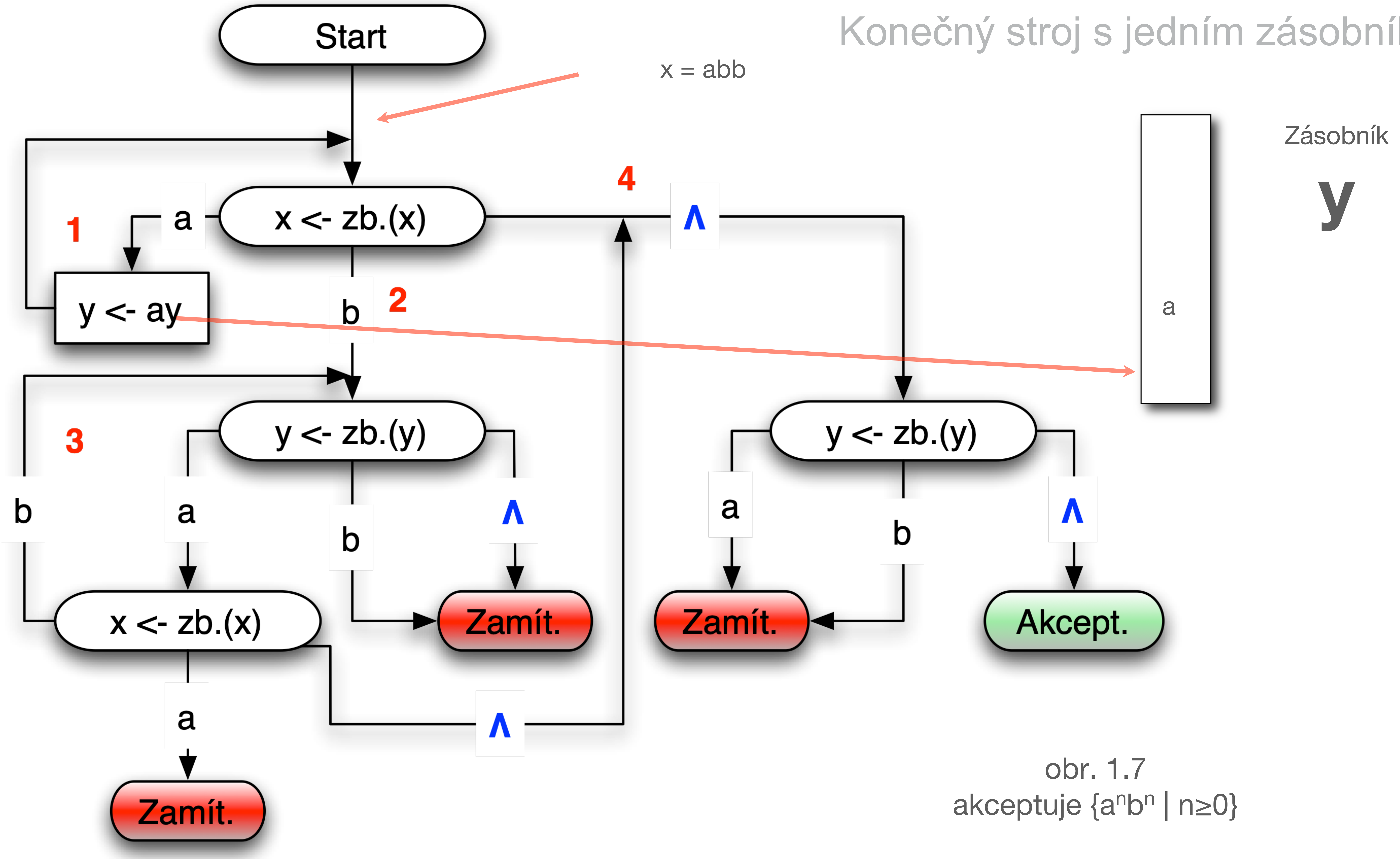
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



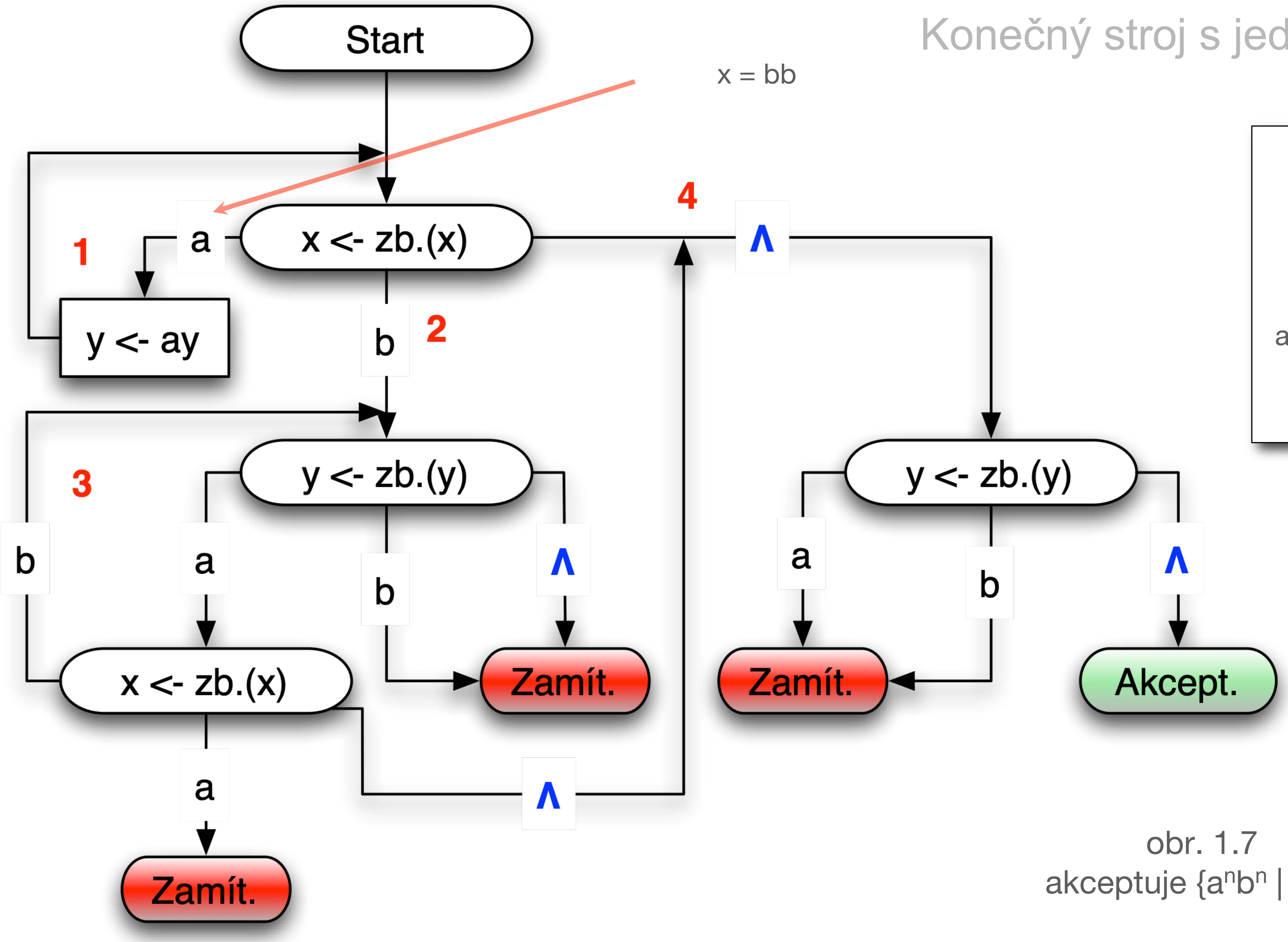
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

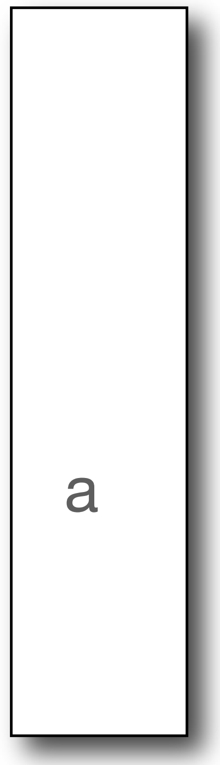
Konečný stroj s jedním zásobníkem



obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



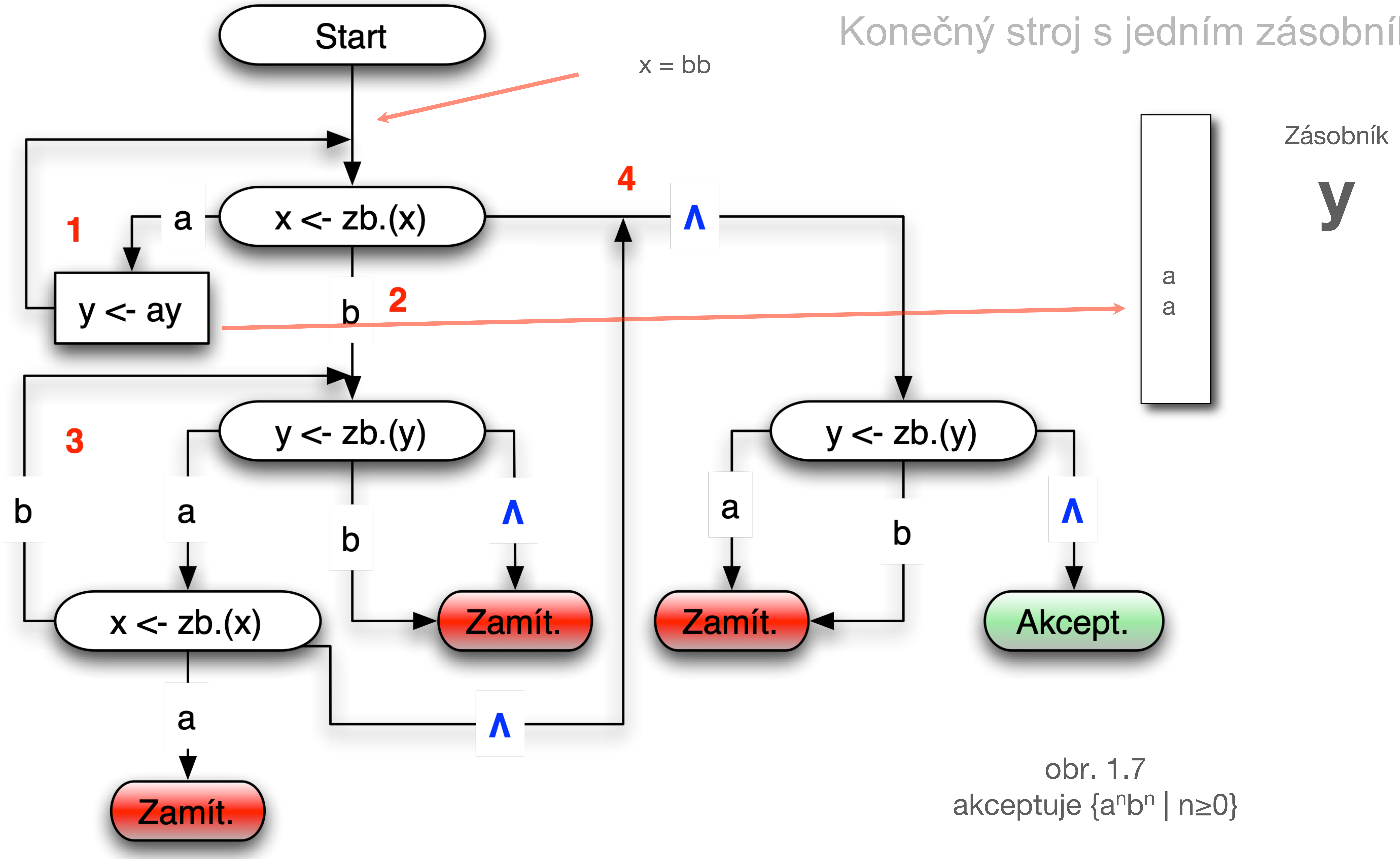


Zásobník

y

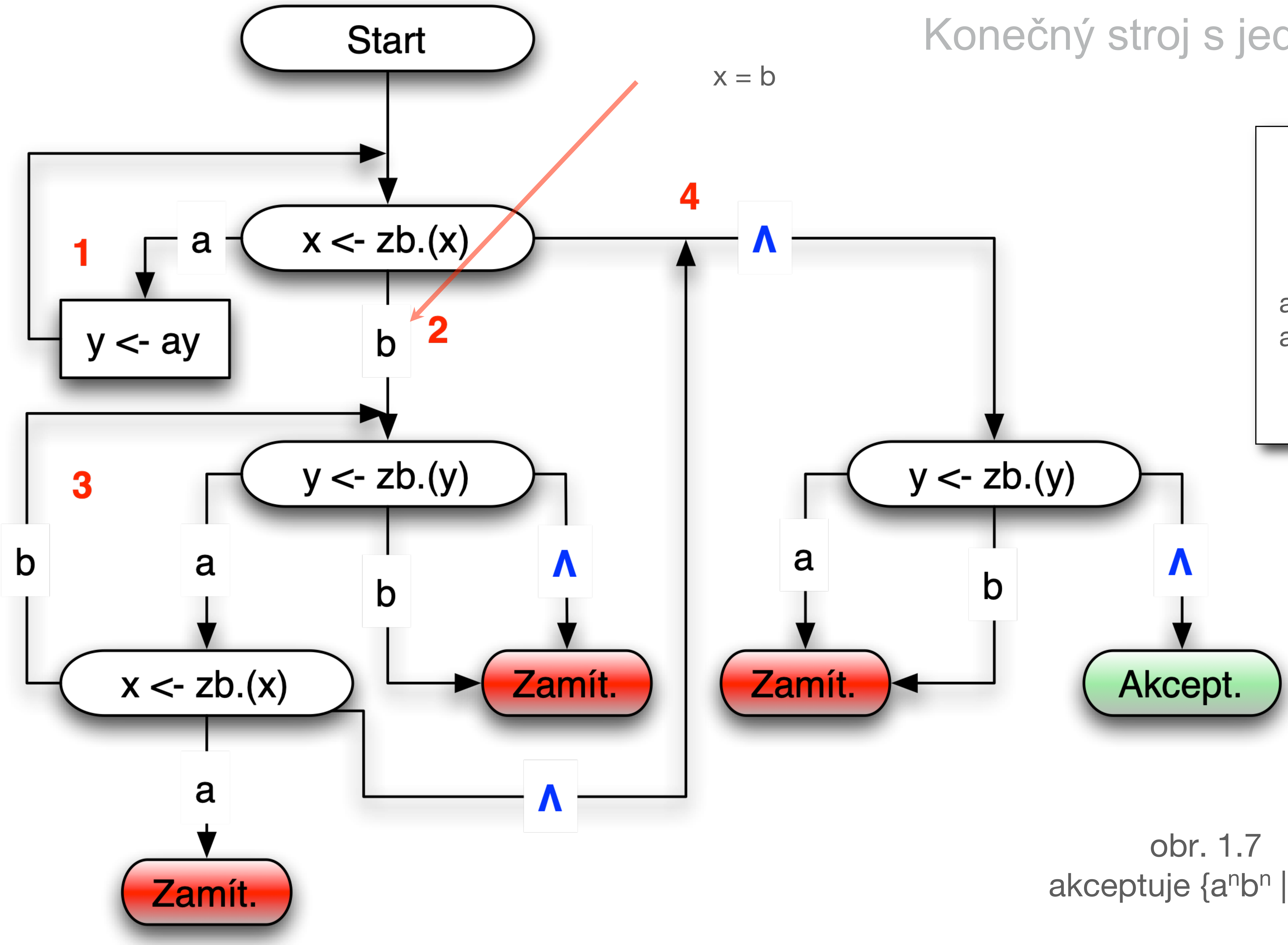
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

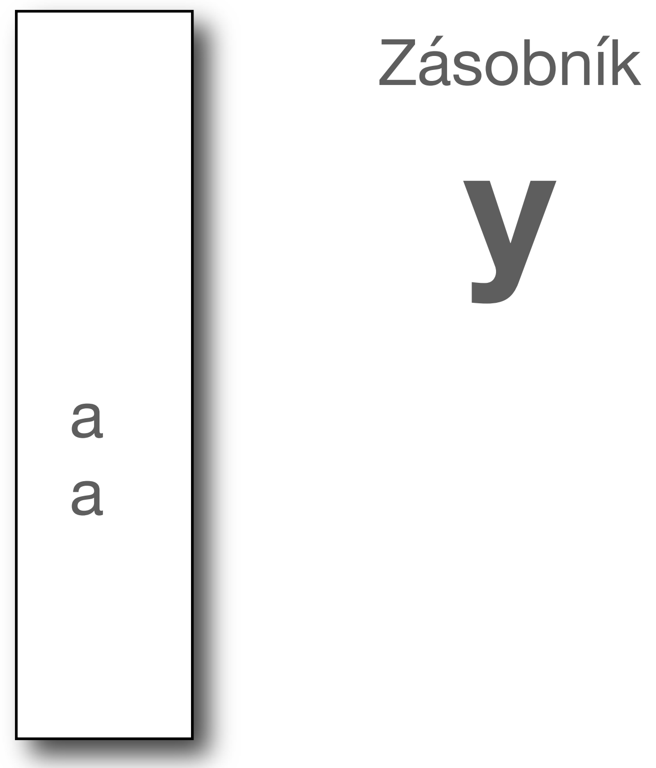
Konečný stroj s jedním zásobníkem



obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

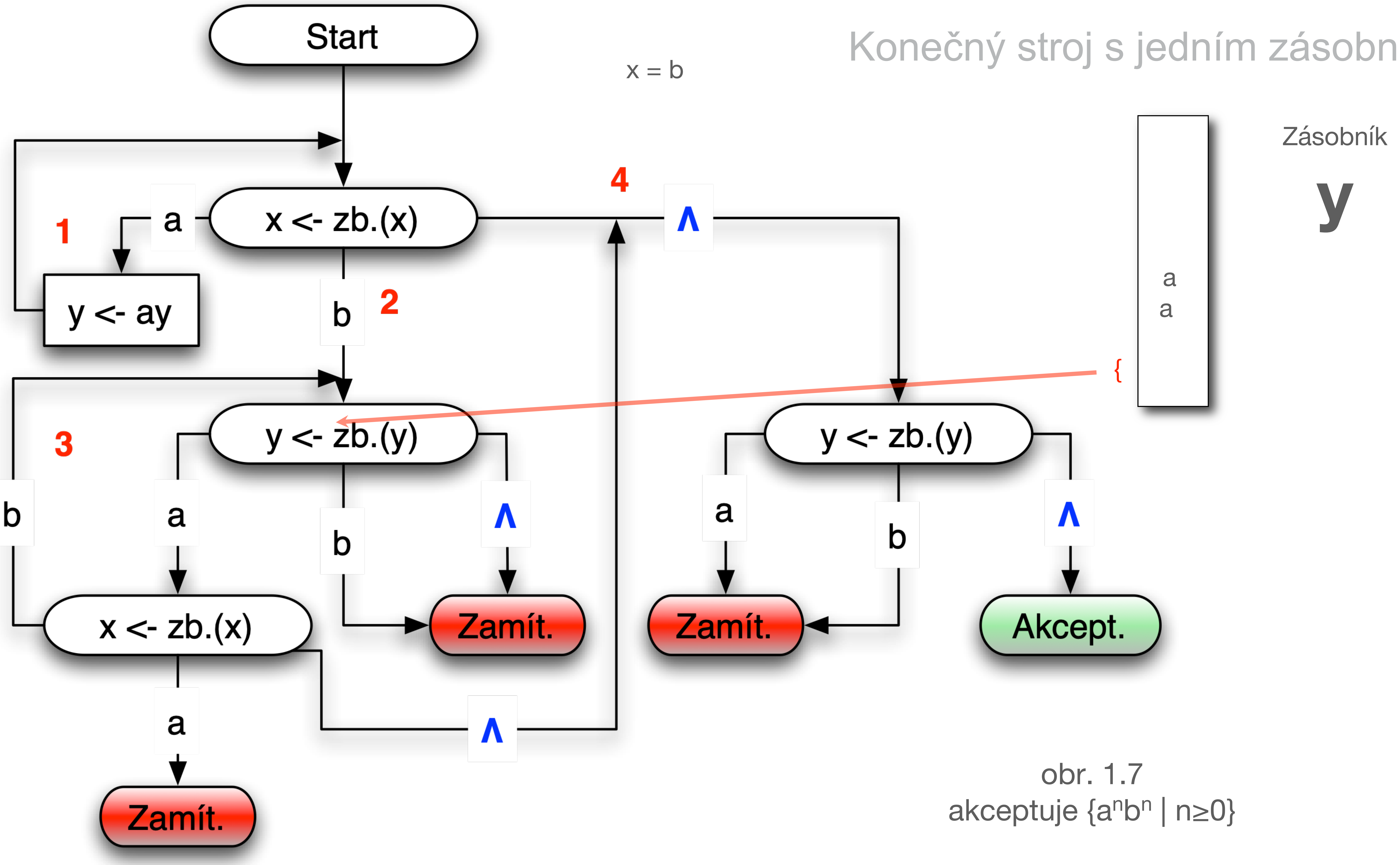
Konečný stroj s jedním zásobníkem





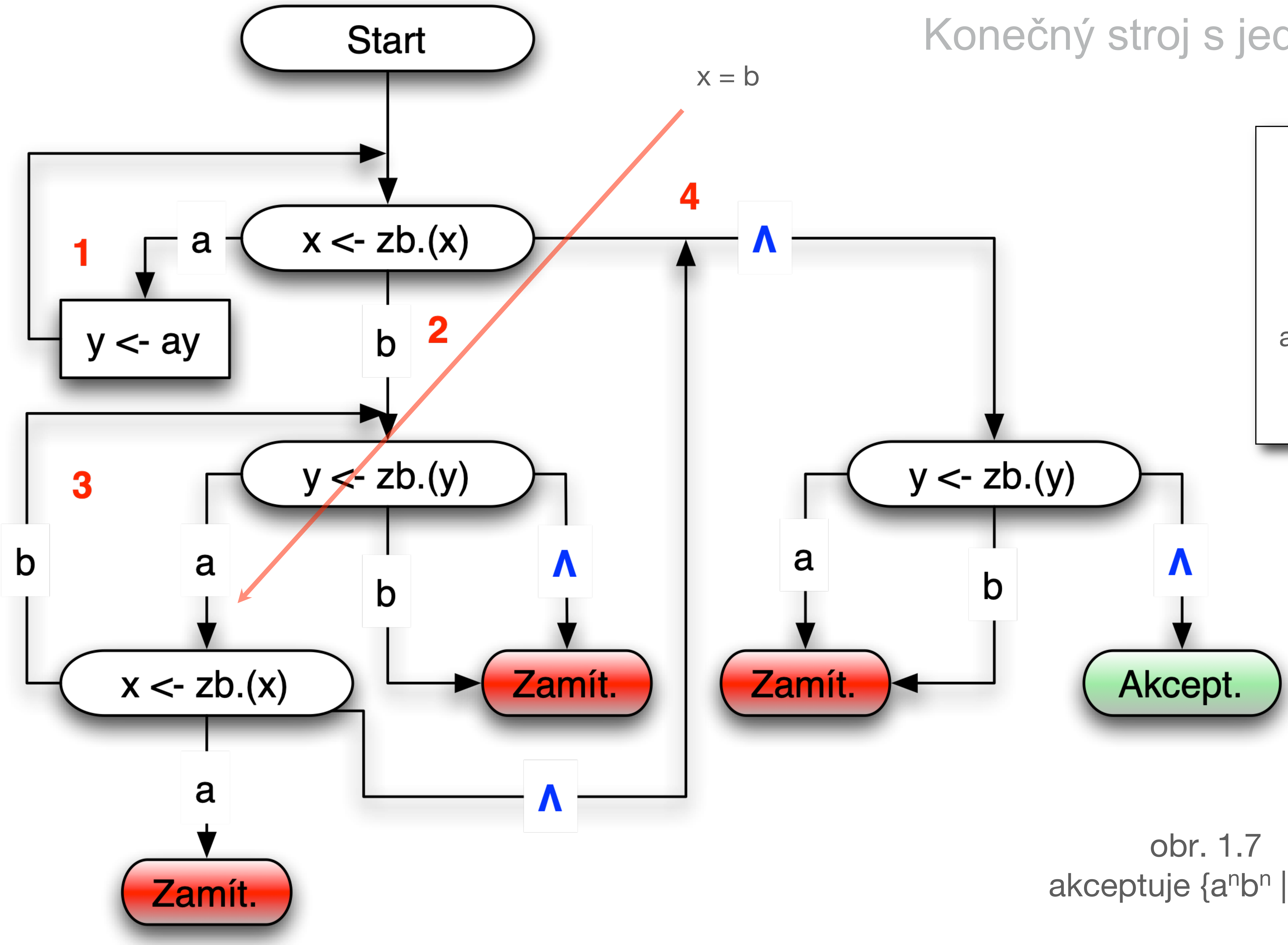
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

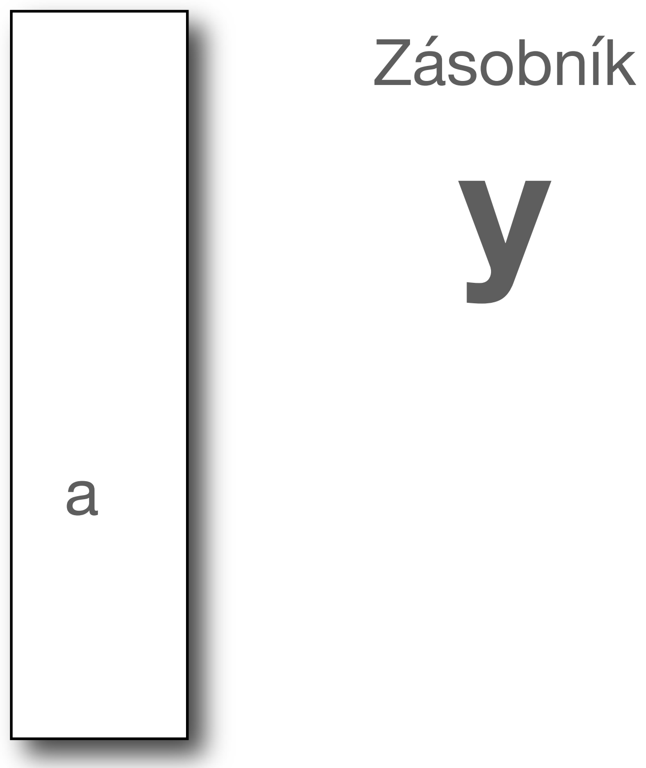
Konečný stroj s jedním zásobníkem



obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

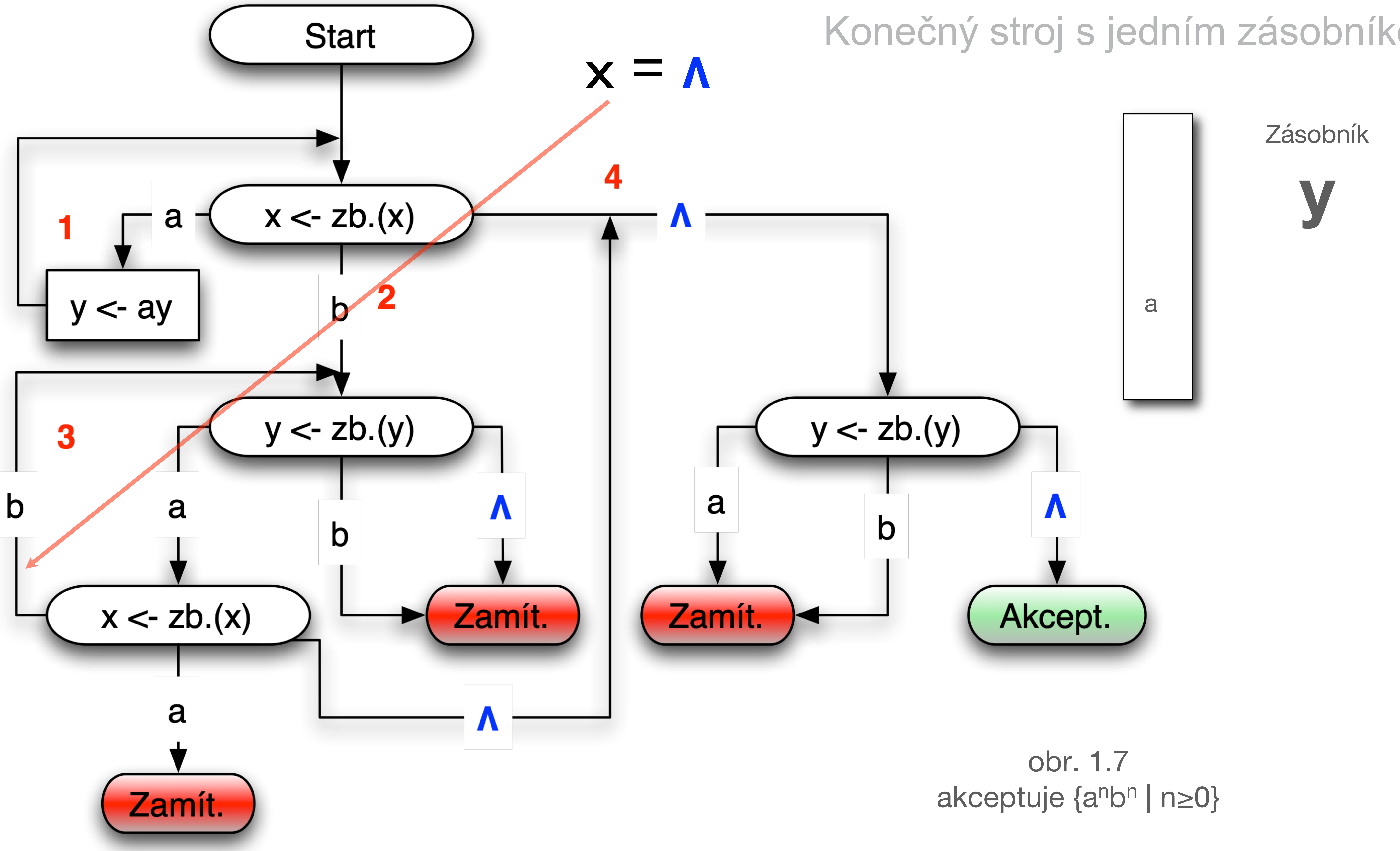
Konečný stroj s jedním zásobníkem





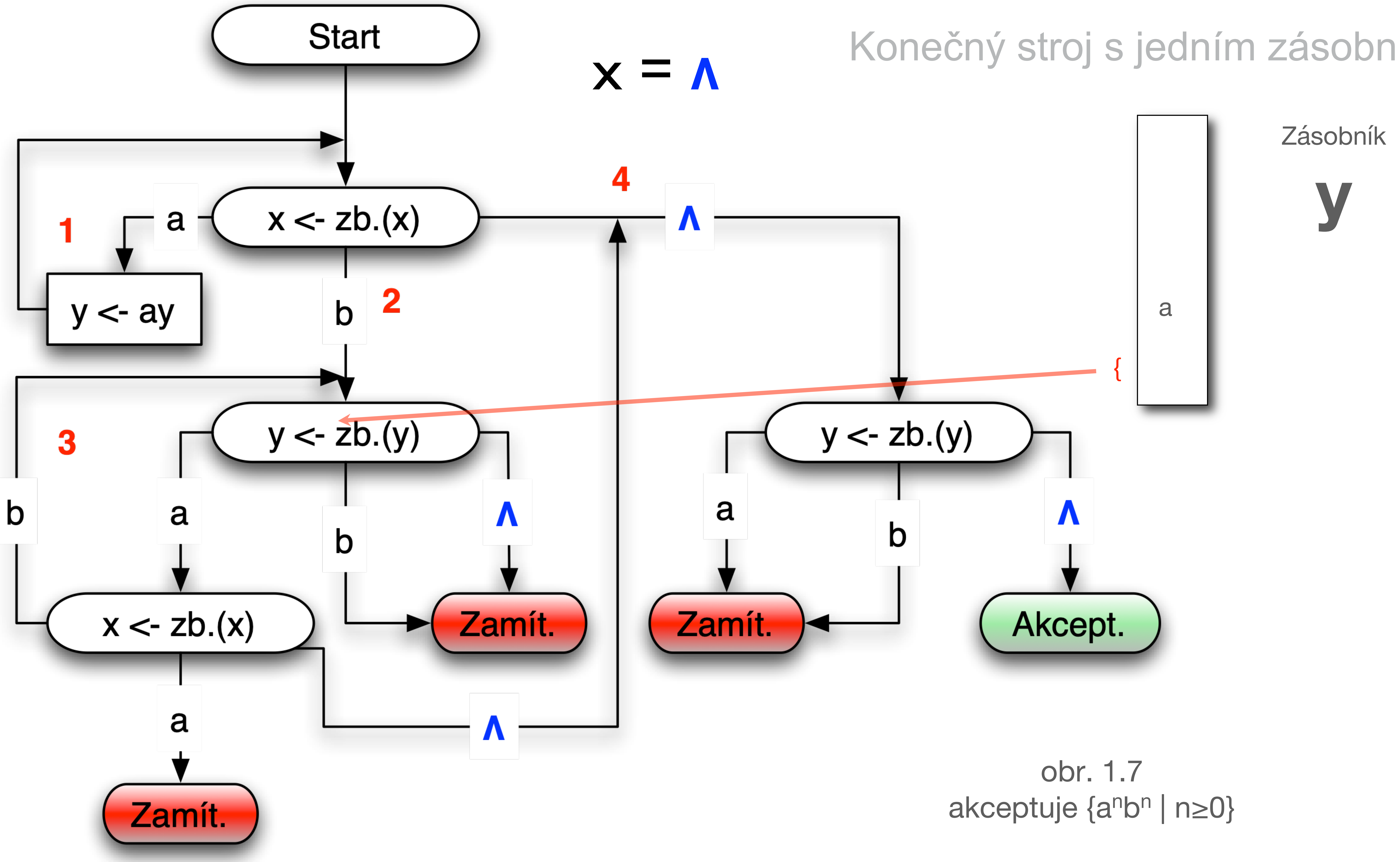
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



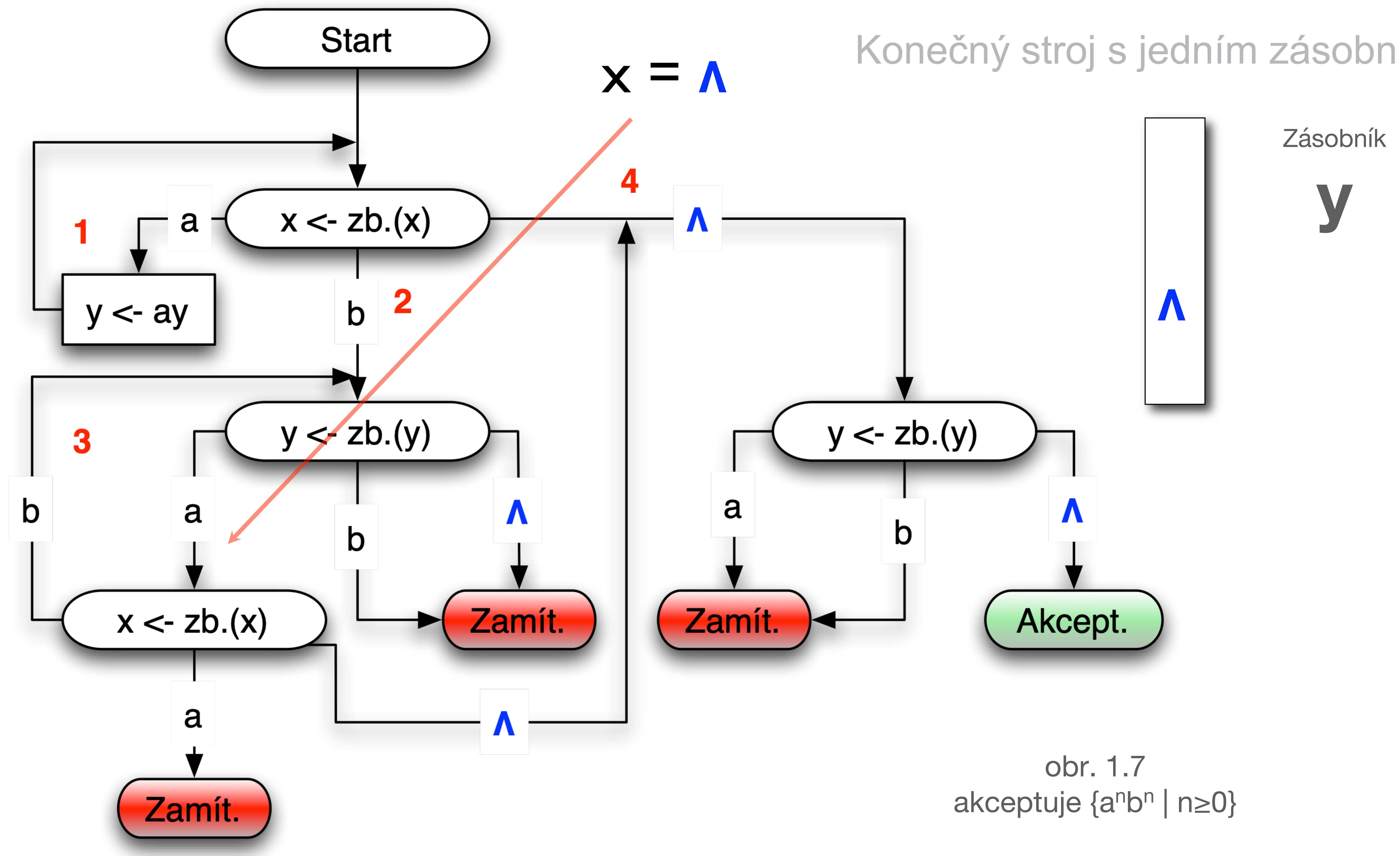
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



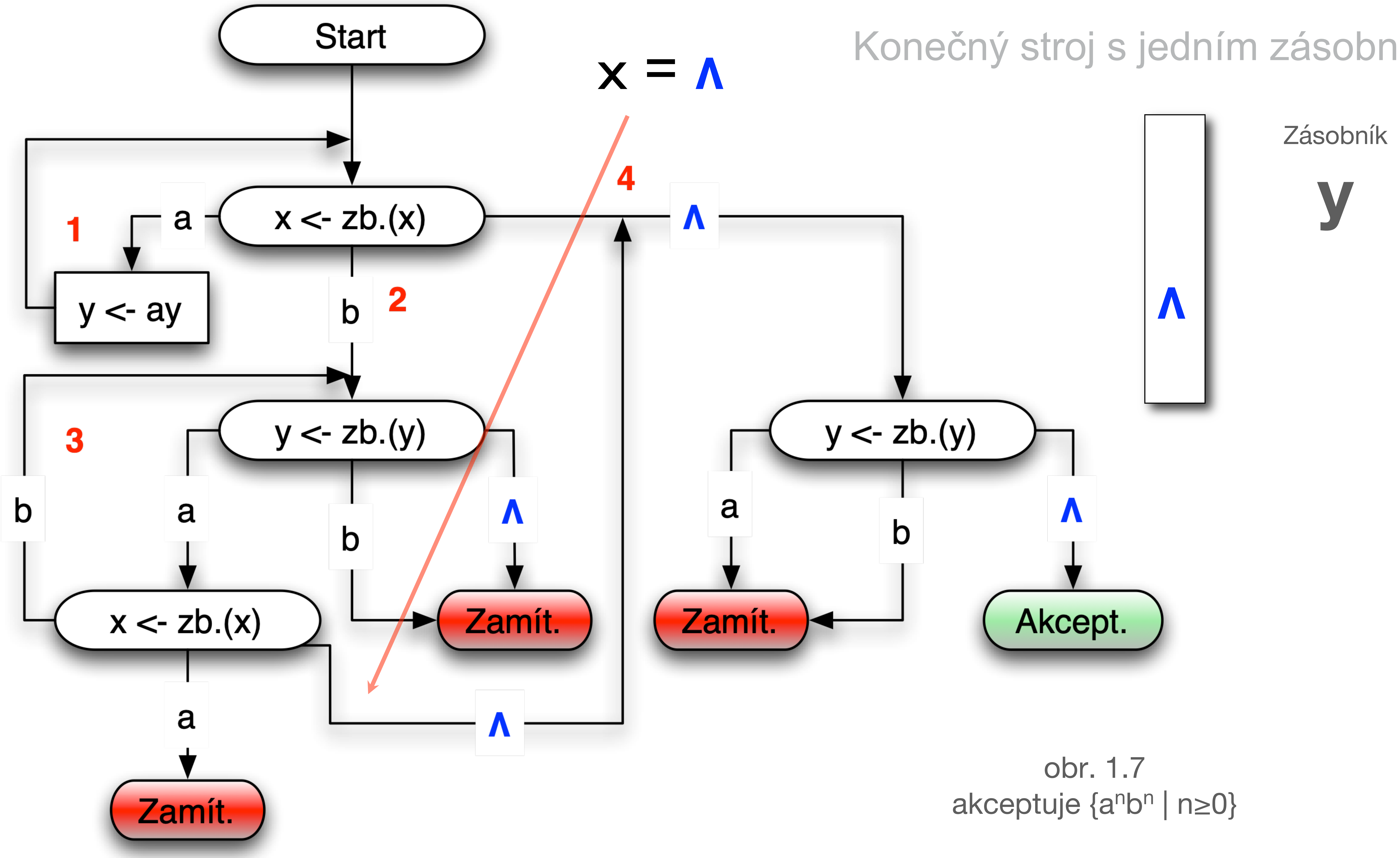
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



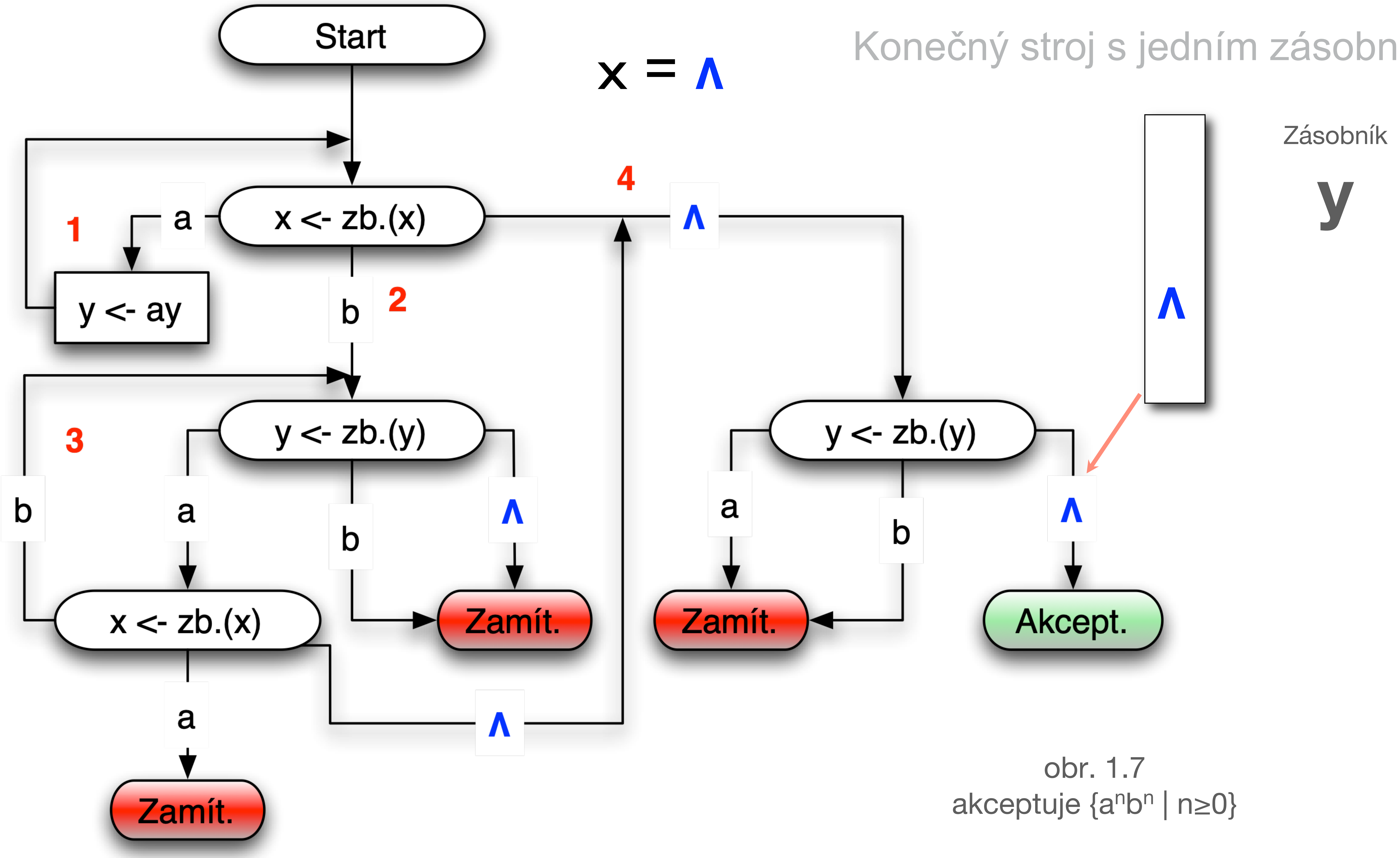
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



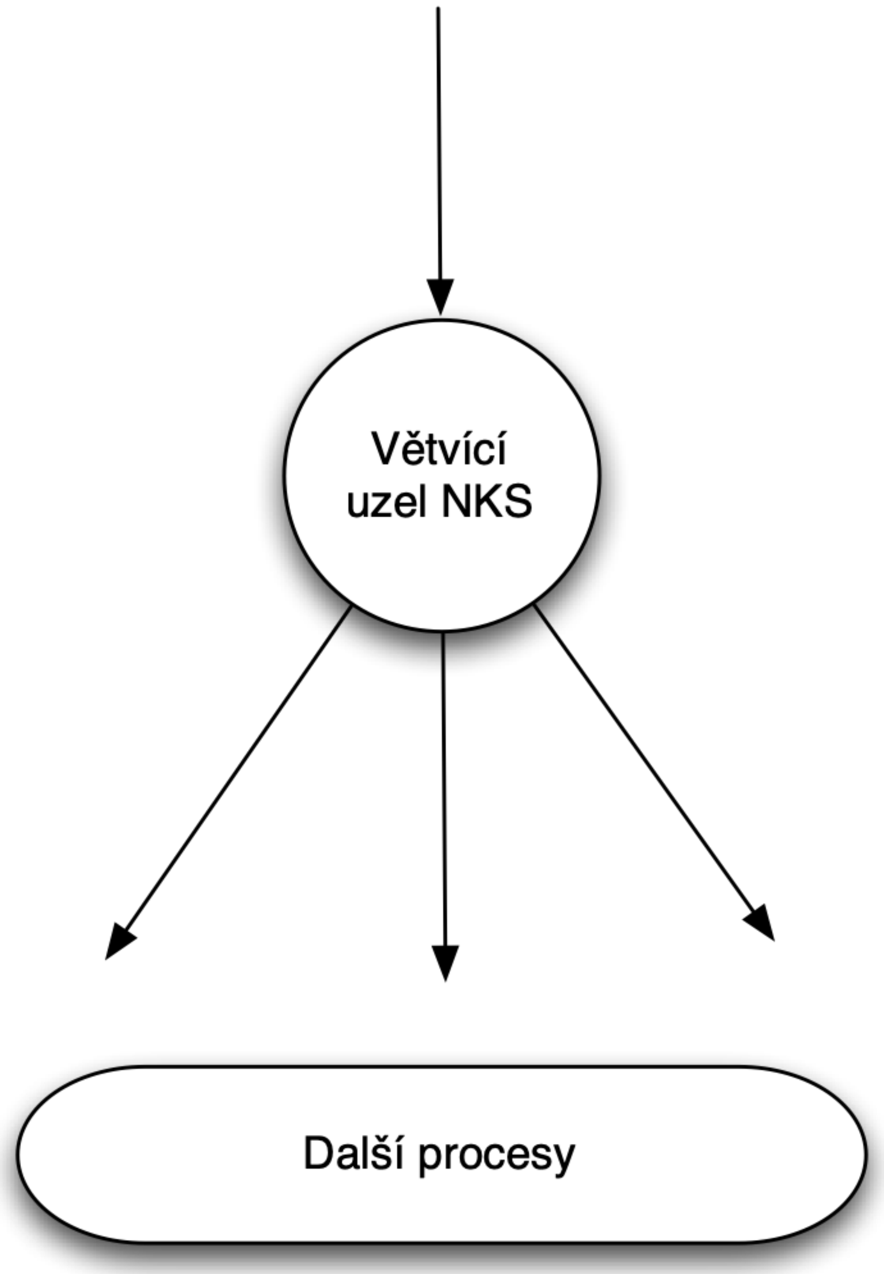
obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Konečný stroj s jedním zásobníkem



obr. 1.7
akceptuje $\{a^n b^n \mid n \geq 0\}$

Nedeterministické KS s (2) ZS

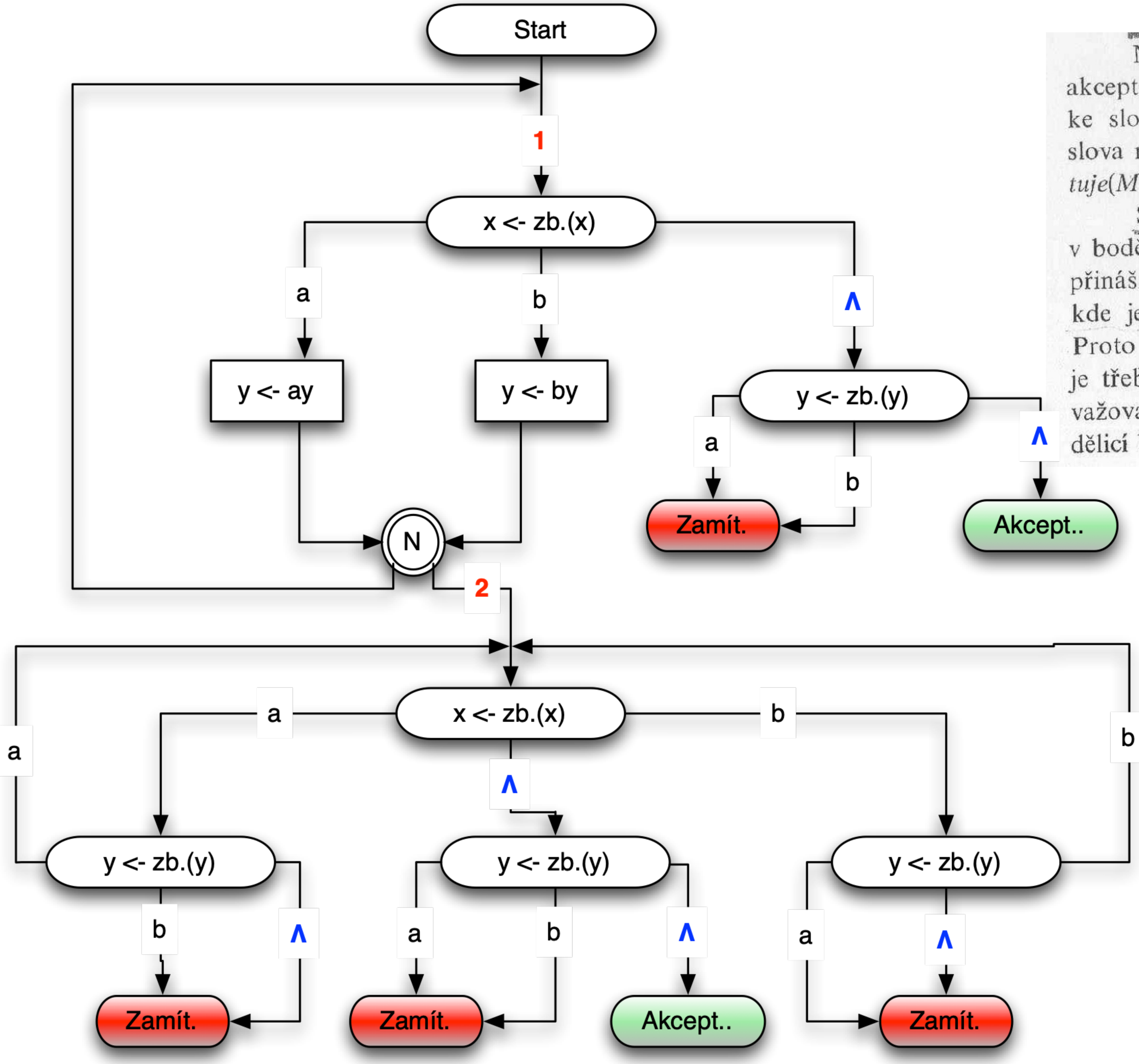


Kdykoliv výpočet dojde k takovému uzlu, může být zvolena kterákoliv z uvažovaných hran jako ta, podle níž bude výpočet dále probíhat. V případě Turingových strojů zavedeme nedeterminismus nikoliv užitím větvících uzlů, ale pouhým odvoláním podmínky, že všechny hrany vycházející z téhož uzlu i musí mít různé hodnoty jmen α.

stejně jako nedeterminismus u konečných autom

Říkáme, že slovo $w \in \Sigma^*$ je *akceptováno* nedeterministickým Turingovým, Postovým nebo konečným strojem M (bez nebo se zásobníky) nad abecedou Σ, existuje-li výpočet začínající se vstupním slovem $x = w$ a končící příkazem AKCEPTOVÁNO.[1]) Není-li w akceptováno, avšak existuje výpočet vedoucí do příkazu ZAMÍTNUTO, je slovo w *zamítnuto*; v ostatních případech $w \in cykluje(M)$.

Nedeterministické KS s (2) ZS



Nedeterministický konečný stroj M_6; *akceptuje*$(M_6) = \{ww^R \mid w \in \{a, b\}^*\}$

Nedeterministický konečný stroj M_6 s jedním zásobníkem y (obr. 1.9) akceptuje všechna slova nad $\Sigma = \{a, b\}$ tvaru ww^R, kde w^R označuje slovo inverzní ke slovu w (tj. slovo s opačným pořadím písmen) a zamítá všechna ostatní slova nad Σ. Je tedy *akceptuje*$(M_6) = \{ww^R \mid w \in \Sigma^*\}$ a *zamítá*$(M_6) = \Sigma^* -$ *akceptuje*(M_6). Množina *akceptuje*(M_6) obsahuje např. slova Λ, *aa*, *babbab*, *bbbabbabbb* atd.

Sledujme výpočet pro $x = ww^R$. Se zapisováním w do zásobníku začneme v bodě 1 a s porovnáváním se slovem w^R v bodě 2. Tato jednoduchá idea však přináší jeden problém: Při postupném zpracování x zleva doprava nevíme, kde je dělicí bod, v němž máme ukončit zápis w a začít porovnávání s w^R. Proto je ve vývojovém diagramu užit větvicí bod. Kdykoliv ho výpočet dosáhne, je třeba uvažovat současně obě možnosti: buď se vrátit zpět k bodu 1 (a považovat další písmena za část slova w), nebo pokračovat k bodu 2 (tj. přejít dělicí bod a začít se zpracováním w^R). ■

Nedeterministické KS s (2) ZS

Vzniká přirozená otázka, zda užití nedeterminismu zvýší sílu uvažované třídy strojů. Dá se ukázat, že třída nedeterministických Turingových strojů, nedeterministických Postových strojů i nedeterministických konečných strojů se dvěma nebo více zásobníky je stejně silná jako třída („obyčejných“) Turingových strojů. Vztah mezi různými třídami konečných strojů nad touž abecedou Σ je dán tímto schématem:

konečné automaty

||

konečné stroje bez zásobníků

||

nedeterministické konečné stroje bez zásobníků

∧

deterministické konečné stroje s jedním zásobníkem

∧

nedeterministické konečné stroje s jedním zásobníkem

∧

konečné stroje se dvěma zásobníky

||

nedeterministické konečné stroje se dvěma zásobníky

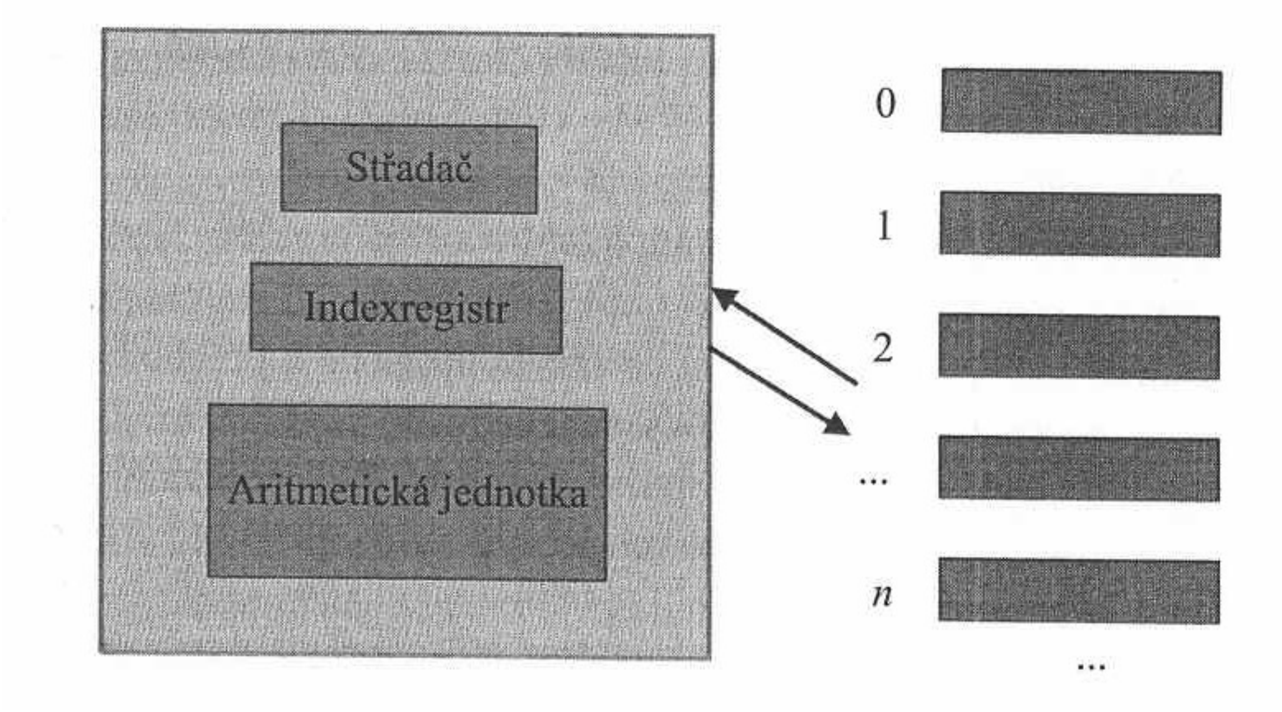
||

Turingovy stroje

Speciálně platí:

1. Neexistuje nedeterministický konečný stroj bez zásobníků, ekvivalentní konečnému stroji M_4 s jedním zásobníkem (př. 1.11).
2. Neexistuje konečný stroj s jedním zásobníkem, ekvivalentní nedeterministickému konečnému stroji M_6 s jedním zásobníkem (př. 1.13).
3. Neexistuje nedeterministický konečný stroj s jedním zásobníkem, ekvivalentní konečnému stroji M_5 se dvěma zásobníky (př. 1.12).

Turingův vs. RASP stroj



- RASP stroj - Random Access Stored Program Machine
- Stroj s pamětí s přímým přístupem
- Stejná síla jako Turingův stroj
- Víceméně podobný Von Neumannově architektuře počítače - pracuje s registry a s řídící jednotkou.
- Velmi omezená množina operačních instrukcí (LOAD, STORE, aritmetické operace a podmíněné a nepodmíněné skoky v závislosti na obsahu střadače a operaci zastavení stroje)
- TS a RASP jsou si ekvivalentní
- RASP je ovšem mnohdy rychlejší (díky přímému adresování paměti - TS musí při čtení dvou vzdálených políček na pásce přečíst i vše mezi nimi)