Prof. Javier Esparza
Philipp Czerner, Martin Helfrich

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2021 – Übungsblatt 5

- Das Übungsblatt ist in zwei Teile gegliedert: den Vorbereitungsteil, den Sie vor der Übung selbstständig bearbeiten sollen, und den Übungs-/Nachbereitungsteil, der Aufgaben enthält, die in der Übung besprochen werden und von Ihnen anschließend zur Nachbereitung verwendet werden können.
- Das ist nicht das Hausaufgabenblatt! Die Hausaufgaben finden Sie auf einem separaten Blatt.

Vorbereitung (→ vor der Übung selbständig zu bearbeiten)

Individualaufgabe Ü5.1. (*Wichtige Begriffe*)

Überprüfen Sie, dass Sie die folgenden Begriffe oder Notationen korrekt definieren können.

- Kontextfreie Sprache (*CFL*)
- Kontextfreie Grammatik (*CFG*)
- rechts-lineare / links-lineare CFG
- (Links-/Rechts-)Ableitung
- Syntaxbaum
- mehrdeutige CFG

Individualaufgabe Ü5.2. (*Automata Tutor: "Contextfree Languages"*)

Lösen Sie die Aufgaben Ü5.2 (a–d) auf Automata Tutor.[1] **Achtung:** Für die Übungsaufgaben haben Sie beliebig viele Versuche. Für die Aufgaben in Hausaufgabe H5.1 nicht!

Lösungsskizze.

(a) $S \rightarrow$ `AundS` $\rightarrow$ `AundA` $\rightarrow$ `AundNVNM` $\rightarrow$ `AundNVNsehr`
$\rightarrow$ `AundNVesparzasehr`
$\rightarrow$ `AundNfeiertesparzasehr`
$\rightarrow$ `Aundstudentfeiertesparzasehr`
$\rightarrow$ `NVNundstudentfeiertesparzasehr`
$\rightarrow$ `NVtheoundstudentfeiertesparzasehr`
$\rightarrow$ `Nbrauchttheoundstudentfeiertesparzasehr`
$\rightarrow$ `tumbrauchttheoundstudentfeiertesparzasehr`

(b) $S \rightarrow YX \rightarrow bYYX \rightarrow baYX \rightarrow baGX \rightarrow badX \rightarrow badaX \rightarrow badaaX \rightarrow badaaS \rightarrow badaaF \rightarrow badaac$

(c) $S \rightarrow SS \mid A$
$A \rightarrow B \mid C \mid D \mid \epsilon$
$B \rightarrow \{C\}$
$C \rightarrow [D] \mid CC \mid DC \mid CD$
$D \rightarrow ()D \mid ()$

[1] Wenn Sie Automata Tutor noch nicht verwendet haben, folgen Sie erst den Schritten in Ü1.2, um sich richtig zu registrieren.

(d) $S \to aSb \mid A \mid B$
$A \to aA \mid a$
$B \to bB \mid b$

Individualaufgabe Ü5.3. (*Anwendungsbeispiel kontextfreie Grammatiken*)

Betrachten Sie die folgende Grammatik $G = (V, \Sigma, P, J)$ mit

$$V = \{J, D, T, N, N', Z, A, S, E, U, B, C, V, U', B'\}$$
$$\Sigma = \{\texttt{;}, \texttt{\{}, \texttt{\}}, \texttt{(}, \texttt{)}, \texttt{=}, \texttt{a}, \texttt{b}, \ldots, \texttt{y}, \texttt{z}, \texttt{0}, \texttt{1}, \ldots, \texttt{8}, \texttt{9}, \texttt{+}, \texttt{-}, \cdot, \texttt{/}, \texttt{\%}, \texttt{!}, \texttt{<}, \texttt{>}, \texttt{\&\&}, \texttt{||}\}$$

mit den Produktionen $P :=$

$$\begin{aligned}
J &\to DS \mid S \\
D &\to TN\texttt{;}D \mid TN\texttt{;} \\
T &\to \texttt{int} \\
N &\to AN' \\
N' &\to AN' \mid ZN' \mid A \mid Z \\
A &\to \texttt{a} \mid \texttt{b} \mid \texttt{c} \mid \ldots \mid \texttt{y} \mid \texttt{z} \\
Z &\to \texttt{0} \mid \texttt{1} \mid \ldots \mid \texttt{8} \mid \texttt{9} \\
S &\to SS \mid \texttt{;} \mid \{S\} \mid N = E\texttt{;} \mid N = \texttt{read();} \mid \texttt{write(}E\texttt{);} \mid \texttt{if(}C\texttt{)}\ S\ \texttt{else}\ S \mid \texttt{while(}C\texttt{)}\ S \\
E &\to Z \mid N \mid (E) \mid UE \mid EBE \\
U &\to \texttt{-} \\
B &\to \texttt{-} \mid \texttt{+} \mid \cdot \mid \texttt{/} \mid \texttt{\%} \\
C &\to \texttt{true} \mid \texttt{false} \mid (C) \mid EVE \mid U'(C) \mid CB'C \\
V &\to \texttt{==} \mid \texttt{!=} \mid \texttt{<=} \mid \texttt{<} \mid \texttt{>=} \mid \texttt{>} \\
U' &\to \texttt{!} \\
B' &\to \texttt{\&\&} \mid \texttt{||}
\end{aligned}$$

(a) Was für eine Sprache erzeugt diese Grammatik?

(b) Beurteilen Sie die folgende Aussagen: Alle Worte in $L(G)$ können zu einem funktionierenden Programm compiliert werden.

(c) Geben Sie ein gültiges Wort in der Sprache an, das alle Nichtterminale mindestens einmal verwendet.

(d) Zeichnen Sie den Syntaxbaum für das in Teilaufgabe (c) gefundene Wort.

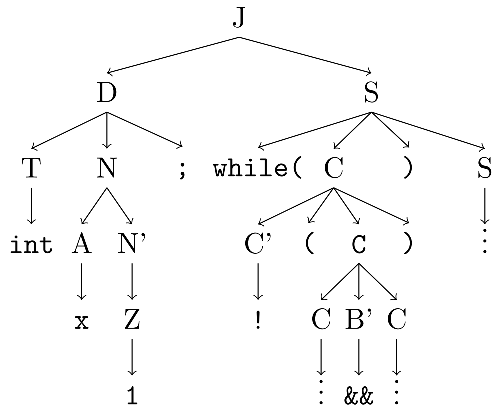
Lösungsskizze.

(a) Alle syntaktisch korrekten MiniJava Programme (siehe Vorlesung: *Einführung in die Informatik 1*)

(b) Die Aussage ist falsch. Die semantische Korrektheit des Programms kann durch die Grammatik nicht sichergestellt werden, nur die syntaktische. Es können zum Beispiel Variablen verwendet werden, die nicht deklariert sind.

(c)
```
int x1;
while(!(x2 > 0))
{ if(x3==x4 && true) write(x5); else x7 = -1 % 5; }
```

(d) Syntaxbaum:



Übung und Nachbereitung

Fokusaufgabe Ü5.4. (*Sprache einer kontextfreien Grammatik*)

Gegeben sei die folgende Grammatik G:

$$S \to aSa \mid bSb \mid aTb \mid bTa$$
$$T \to aT \mid bT \mid \epsilon$$

(a) Welche Sprache beschreibt G? Geben Sie eine intensionale Mengendarstellung[2] L für $L(G)$ an.

(b) Zeigen oder widerlegen Sie: L ist regulär.

(c) Zeigen Sie $L(G) = L$ formal. Beweisen Sie dabei auch induktiv, welche Sprache von T erzeugt wird.

Übungsaufgabe Ü5.5. (*Sprache einer kontextfreien Grammatik*)

Sei $G = (\{S, E, O, A, B, X\}, \{a, b\}, P, S)$ die CFG mit folgenden Produktionen P:

$$S \to E \mid O$$
$$E \to AB \mid BA$$
$$A \to XAX \mid a$$
$$B \to XBX \mid b$$
$$O \to XXO \mid X$$
$$X \to a \mid b$$

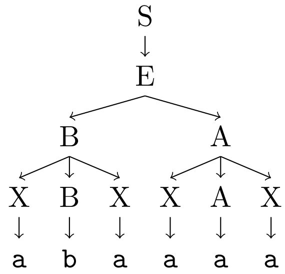
(a) Geben Sie für jedes der folgenden Wörter jeweils eine Linksableitung und eine Rechtsableitung zzgl. des entsprechenden Syntaxbaums an:
(i) *abaaaa* (ii) *babab* (iii) *aabbaaba*

(b) Entscheiden Sie, ob die Grammatik G mehrdeutig ist oder nicht. Wenn sie mehrdeutig ist, geben Sie ein Wort $w \in L(G)$ mit zwei Syntaxbäumen an. Sonst beweisen Sie, dass G nicht mehrdeutig ist.

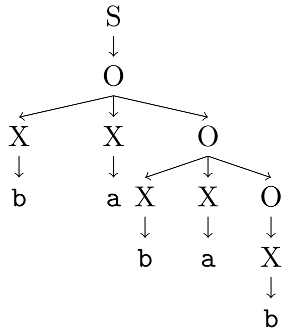
[2] Das heißt, eine Beschreibung der Form $L := \{w \in A \mid P(a)\}$ für eine geeignete Menge A und Prädikat P.

Lösungsskizze.

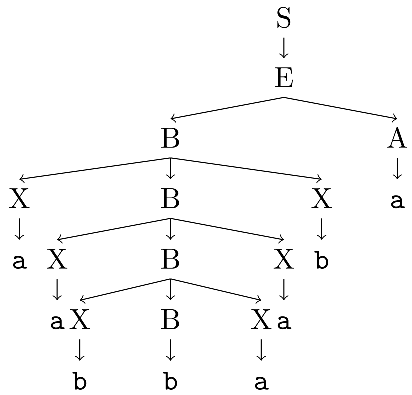
(a) (i)
Linksableitung: $S \to E \to BA \to XBXA \to aBXA \to abXA \to abaA \to abaXaX \to abaaaX \to abaaaa$
Rechtsableitung: $S \to E \to BA \to BXaX \to BXaa \to Baaa \to XbXaaa \to Xbaaaa \to abaaaa$



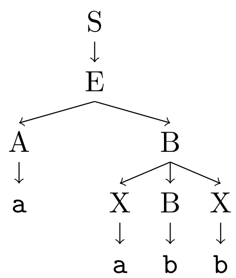
(ii)

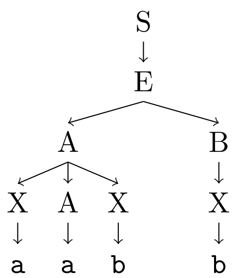


(iii)



(b) G ist nicht eindeutig:





Übungsaufgabe Ü5.6. (*Residualsprachen*)

Die Äquivalenzklassen einer Sprache L kann man verwenden um z.B. direkt den minimalen DFA aufzustellen, oder um festzustellen, ob L regulär ist. Allerdings ist es teilweise mühsam, die Äquivalenzklassen herauszufinden: die Äquivalenzklasse von u enthält alle Wörter v, sodass die Suffixe w, die man an u anhängen muss, damit $uw \in L$, genau die sind, die man an v anhängen muss, damit $vw \in L$. Wir betrachten nun diese Suffixe direkt und zeigen, dass man auch auf diese Weise bereits den minimalen DFA aufstellen kann und entscheiden kann, ob eine Sprache regulär ist.

Sei $\Sigma := \{a, b\}$ und $L \subseteq \Sigma^*$ eine Sprache über Σ. Für ein Wort $w \in \Sigma^*$ definieren wir die *Residualsprache* $L^w := \{u : wu \in L\}$. Die Residualsprache L^w enthält also genau die Wörter in L, die mit w beginnen, wobei das führende w entfernt wurde. Beispielsweise gilt $L(ab^* \,|\, ba^*)^a = L(b^*)$ und $L(ab^* \,|\, ba^*)^{aba} = \emptyset$.

(a) Zeigen Sie $u \equiv_L v \Leftrightarrow L^u = L^v$ für beliebige Wörter $u, v \in \Sigma^*$.

(b) Folgern Sie aus der (a), dass L genau dann regulär ist, wenn L endlich viele (unterschiedliche) Residualsprachen besitzt, also $|\{L^w : w \in \Sigma^*\}| < \infty$.

(c) Konstruieren Sie den kanonischen Minimalautomaten zu dem regulären Ausdruck $r := ab \,|\, ba^*$, indem Sie einen Zustand $[w]_{\equiv_{L(r)}}$ nicht mit dessen Äquivalenzklasse, sondern mit einem regulären Ausdruck für die Residualsprache $L(r)^w$ beschriften.

Lösungsskizze. **(a)** Seien $u, v \in \Sigma^*$ beliebig. Wir teilen den Beweis in zwei Richtungen auf.

„$\Rightarrow$“ Es gelte also $u \equiv_L v$. Zur Erinnerung bedeutet dies, dass für alle $w \in \Sigma^*$ gilt, dass $uw \in L \Leftrightarrow vw \in L$. Wir müssen für alle Wörter $w \in \Sigma^*$ zeigen, dass $w \in L^u \Leftrightarrow w \in L^v$. Sei also w beliebig. Es gilt

$$w \in L^u \overset{(1)}{\iff} uw \in L \overset{(2)}{\iff} vw \in L \overset{(1)}{\iff} w \in L^v$$

Bei (1) verwenden wir die Definition von Residualsprachen, bei (2) benutzen wir $u \equiv_L v$, wie oben angemerkt.

„$\Leftarrow$“ Nun nehmen wir $L^u = L^v$ an. Es ist zu zeigen, dass für alle Wörter $w \in \Sigma^*$ gilt, dass $uw \in L \Leftrightarrow vw \in L$. Es gilt für alle w:

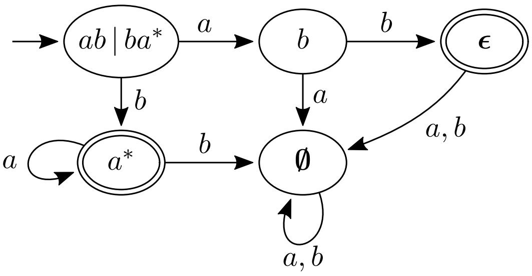
$$uw \in L \overset{(1)}{\iff} w \in L^u \overset{(2)}{\iff} w \in L^v \overset{(1)}{\iff} vw \in L.$$

Bei (1) verwenden wir wieder die Definition von Residualsprachen, bei (2) $L_u = L_v$.

(b) Aus (a) folgt $u \not\equiv_L v \Leftrightarrow L^u \neq L^v$. Das bedeutet, die Anzahl Äquivalenzklassen bezüglich $\equiv_L$ ist genau die Anzahl der verschiedenen Residualsprachen. Laut Myhill-Nerode (Satz 3.60) ist eine Sprache regulär genau dann, wenn sie endlich viele Äquivalenzklassen besitzt. Aufgrund der eben festgestellten Gleichheit also regulär genau dann, wenn sie endlich viele Residualsprachen besitzt.

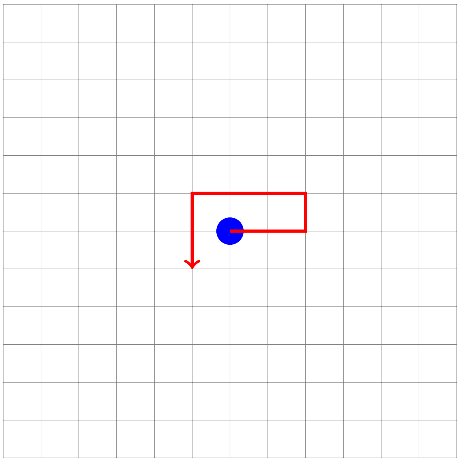
Etwas formaler sei $M_1 := \{[w]_{\equiv_L} : w \in \Sigma^*\}$ die Menge der Äquivalenzlassen von $\equiv_L$ und $M_2 := \{L^w : w \in \Sigma^*\}$ die Menge der Residualsprachen von L. Wir definieren die Funktion $f : M_1 \to M_2$ als $f([w]_{\equiv_L}) := L^w$ für alle $w \in \Sigma^*$. Nach Teilaufgabe (a) hängt $f([w]_{\equiv_L})$ nicht davon ab, welchen Repräsentanten $w \in [w]_{\equiv_L}$ wir wählen, f ist also wohldefiniert. Für Äquivalenzklassen $[u]_{\equiv_L} \neq [v]_{\equiv_L}$ gilt $L^u \neq L^v$ (wieder nach (a)), und somit $f([u]_{\equiv_L}) \neq f([v]_{\equiv_L})$. Also ist f injektiv. Da jedes Wort $w \in \Sigma^*$ eine Äquivalenzklasse $[w]_{\equiv_L}$ besitzt, ist f auch surjektiv. Also ist f eine Bijektion und

$|M_1| = |M_2|$ gilt; insbesondere gibt es genau dann unendlich viele Äquivalenzklassen, wenn es unendlich viele Residualsprachen gibt.
(c) Wir erhalten folgenden Automaten:



Übungsaufgabe Ü5.7. (*Pfeilsprachen*)

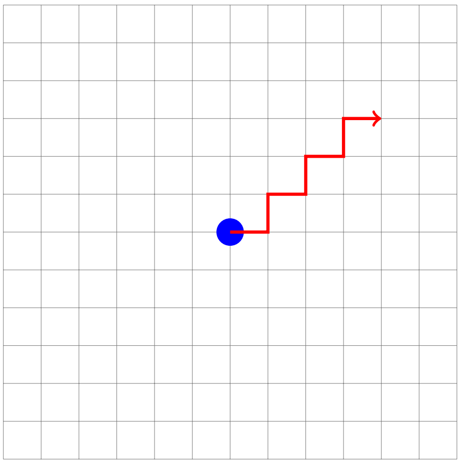
In dieser Aufgabe betrachten wir Sprachen, deren Worte Linienzüge in einem unendlichen zweidimensionalen Gitter von einem fixen Startpunkt aus beschreiben. Die folgende Grafik zeigt einen Ausschnitt aus dem Gitter:

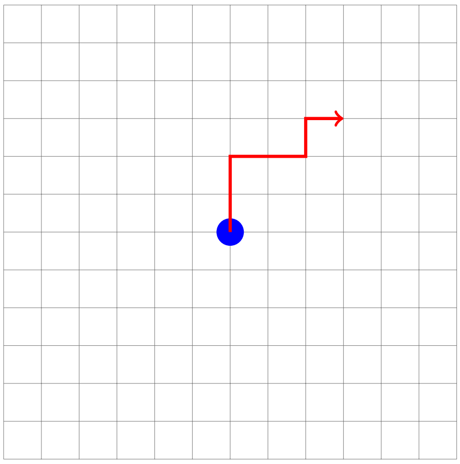


Wir haben Startpunkt blau markiert. Linienzüge beschreiben wir im Folgenden als eine Sequenz von Pfeilen, d.h. als Worte über dem Alphabet $\Sigma = \{\rightarrow, \leftarrow, \uparrow, \downarrow\}$. Die Pfeile beschreiben dabei (vom Startpunkt aus gesehen) einen ein Kästchen langen Schritt entlang des Gitters. Wir stellen daher den im Bild rot eingezeichnete Linienzug durch das Wort $w = \rightarrow\rightarrow\uparrow\leftarrow\leftarrow\leftarrow\downarrow\downarrow$ dar.

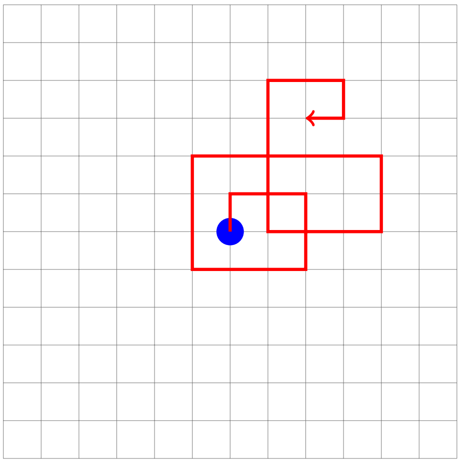
(a) Betrachten Sie die folgenden natürlich sprachlichen Beschreibungen zusammen mit jeweils einem Beispiel, welches in der Sprache liegt (auf der linken Seite), und einem Beispiel, das kein Element der Sprache ist (auf der rechten Seite). *Geben Sie für jede der Sprachen eine formale Definition der Form* $\{w \in \Sigma^* \mid \ldots\}$ *an.*[3]

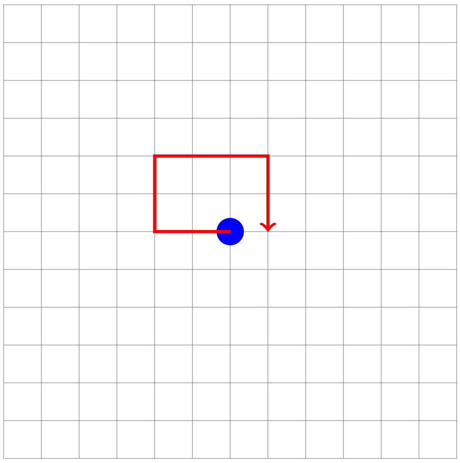
(1) die Sprache aller Treppen über dem Alphabet $\Sigma' = \{\rightarrow, \uparrow\}$





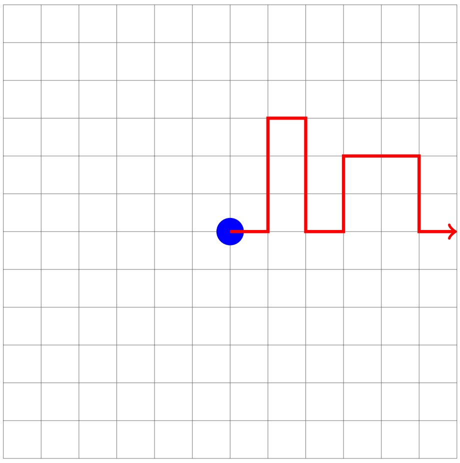
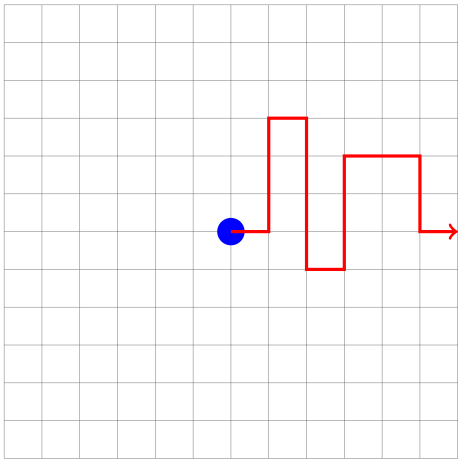
(2) die Sprache aller im Uhrzeigersinn laufenden Spiralen über dem Alphabet Σ, die vom Startpunkt aus zuerst nach oben laufen





(3) die Sprache aller "Skylines" über dem Alphabet $\Sigma'' = \{\rightarrow, \uparrow, \downarrow\}$

[3] Das heißt insbesondere, dass Sie in diesem Aufgabenteil keinen Automaten, keinen regulären Ausdruck, keine Grammatik oder ähnliches angeben sollen, die die Sprache beschreiben.

Hinweis: Die Sprachen sind mithilfe der Beispiele nicht eindeutig bestimmt! Ziel der Aufgabe ist es, die intuitive Beschreibung (z.B. “Sprache aller Skylines”) zusammen mit den Beispielen in eine möglichst allgemeine Sprachdefinition zu bringen.

(b) Stellen Sie Vermutungen auf, ob die obigen Sprachen regulär oder kontextfrei sind. Begründen Sie Ihre Antwort möglichst anschaulich anhand des Beispiels.

(c) Geben Sie zu jeder der Sprachen L aus Aufgabenteil (a) eine Grammatik G an.

Lösungsskizze.

(a) (1) $L = \{(\rightarrow\uparrow)^i \mid i \in N\}\{\varepsilon, \rightarrow\}$

(2) $L = \{u \in \Sigma^* \mid \exists v \in \Sigma^*.\ uv \in L((\uparrow^+\rightarrow^+\downarrow^+\leftarrow^+)^*)\}$

(3) $L = \{w \in \Sigma^* \mid (\forall u, v \in \Sigma^*.\ w = uv \rightarrow |u|_\uparrow \geq |u|_\downarrow) \wedge (\forall u, v \in \Sigma^*.\ \forall x, y \in \Sigma.\ w = uxyv \rightarrow xy \notin \{\uparrow\downarrow, \downarrow\uparrow\})\}$

(b) (1) regulär, da nur eine Alternierung konstanter Pfade verlangt wird.

(2) regulär, da nur die Laufrichtung wichtig ist, aber nicht die Länge.

(3) kontextfrei, da zwei Längen verglichen werden müssen.

(c) (1) $S \mapsto \rightarrow\uparrow S \mid \rightarrow \mid \varepsilon$

(2) $S \mapsto \uparrow S \mid \uparrow T \mid \varepsilon \qquad T \mapsto \rightarrow T \mid \rightarrow U \mid \varepsilon \qquad U \mapsto \downarrow U \mid \downarrow V \mid \varepsilon \qquad V \mapsto \leftarrow V \mid \leftarrow S \mid \varepsilon$

(3) $S \mapsto T_? \mid T_? \rightarrow S \qquad T_? \mapsto \varepsilon \mid T \qquad T \mapsto \uparrow T \downarrow \mid T_? \rightarrow T_?$