Informatik I WS 07/08

Tutorium 24

15.11.07

Bastian Molkenthin

E-Mail: infotut@sunshine2k.de

Web: http://infotut.sunshine2k.de

Universität Karlsruhe (TH)
Forschungsuniversität · gegründet 1825

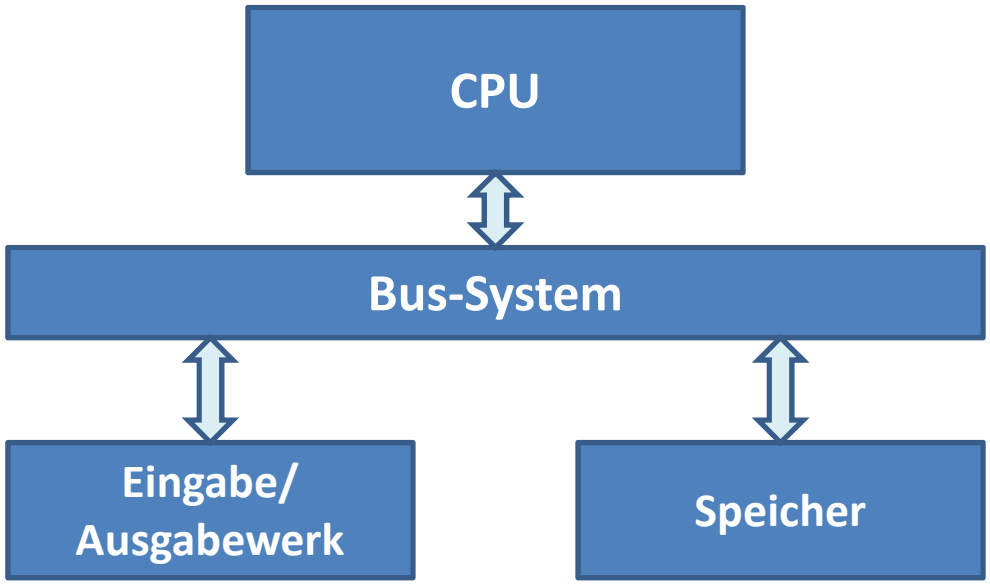
Review / Organisatorisches

- Probleme bei Foliendownload?
- Ab nächste Woche Rechnerübungen! Bitte per E-Mail anmelden.
- Nächste Woche nur 1 Übung da Aufgabe winzig ist… Wann?
- Aufgabe bis spätestens 3 Wochen nach Abgabe vorführen.
- Mittlere Codewortlänge = $\sum_{a_i \in A} p(a_i) * Codelänge(a_i)$
- Sich jetzt schon mit Java befassen
  -> ab nächster Woche verstärkt Praxisaufgaben

„Programmieren lernt man durch programmieren!“

von-Neumann Architektur

Aus welchen Komponenten besteht ein von-Neumann Rechner?



Ein von-Neumann-Rechner besteht aus der Zentraleinheit (CPU, auch Prozessor) des Rechners, dem Speicher zur Aufbewahrung von binären Programm- und Dateninformationen, der Verbindungseinrichtung (Bus) sowie Ein- und Ausgabeeinheiten.

von-Neumann Architektur (2)

Beschreiben Sie das Problem des so genannten *von-Neumann'schen Flaschenhalses*.

- Das Problem des so genannten von-Neumann'schen Flaschen-halses resultiert aus physikalischen Gegebenheiten.
- Der Rechnerkern kann heute Befehle um etwa den Faktor 10 schneller ausführen als Speicherzugriffe.
- Somit wird das Verbindungssystem zum Engpass zwischen Speicher und CPU

Bits & Bytes

- Zahlen besitzen generell Stellenwertigkeiten:
  $123 = 1 * 100 + 2 * 10 + 3 * 1 = 1*10^2 + 2*10^1 + 3 * 10^0$
- Dezimal Zahlen besitzen Basis 10
- Dual Zahlen besitzen Basis 2

⇒ Umrechnung einer Binärzahl ins Dezimalsystem:

$b_i b_{i-1} \ldots b_1 b_0$ sei eine Binärzahl wobei die b_j jeweils den Wert 1 oder 0 annehmen.
Dann ist $z = b_i * 2^i + b_{i-1} * 2^{i-1} + \ldots + b_1 * 2^1 + b_0 * 2^0$
die entsprechende Dezimalzahl.

Aufgabe: Berechne den Dezimalwert von 110101!

Bits & Bytes (2)

➡ Umrechnung einer Dezimalzahl x ins Binärsystem:
z.B. mit Hornerschema

Setze $z_0 = x$.
Berechne $z_i = z_{i-1} / 2$ bis $z_i = 0$

Der Rest jeder Division gibt ein weiteres Bit b_i .
$b_n \ldots b_2 b_1$ ergibt den Wert im Dualsystem

Aufgabe: Berechne den Binärwert von 19!

19 / 2 = 9 Rest 1
9 / 2 = 4 Rest 1
4 / 2 = 2 Rest 0
2 / 2 = 1 Rest 0
1 / 2 = 0 Rest 1

19_{10} = (000)1 0011$_2$

Bits & Bytes (3)

Umrechnung einer Dezimalzahl x ins Binärsystem:
z.B. nach Euklidschem Algorithmus

Finde kleinstes k, für dass $2^k < x < 2^{k+1}$
Berechne $b_i := x_i / 2^{(k-i)}$ für i = 0…k, wobei $x_0 = x$ und
x_i der Rest der vorhergehenden Division ist:

$$x_i := x_{i-1} \text{ modulo } 2^k$$

$b_0 b_1 \ldots b_i$ ergibt den Wert im Dualsystem

Aufgabe: Berechne den Binärwert von 30!

$2^4 = 16 < 30 < 32 = 2^5$
$b_0 = 30/2^4 \quad = 1,\ldots$ Rest 14
$b_1 = 14/2^3 \quad = 1,\ldots$ Rest 6
$b_2 = 6/2^2 \quad = 1,\ldots$ Rest 2
$b_3 = 2/2^1 \quad = 1,\ldots$ Rest 0
$b_4 = 0/2^0 \quad = 0,\ldots$

30_{10} = (000)1 1110_2

Fehlererkennung

Was ist ein Paritätsbit ?

Das Paritätsbit ergänzt die Anzahl der „1“ in der Binärzahl auf eine gerade Anzahl, d.h.

- Anzahl der „1“ gerade — Paritätsbit = 0
- Anzahl der „1“ ungerade — Paritätsbit = 1

Aufgabe: Gib das Paritätsbit von 11001011_2 und 14_{10} an!

11001011_2 — Paritätsbit = 1

$14_{10} = 1110_2$ — Paritätsbit = 1

Aufgabe: Ist im folgenden Fall ein Fehler aufgetreten?

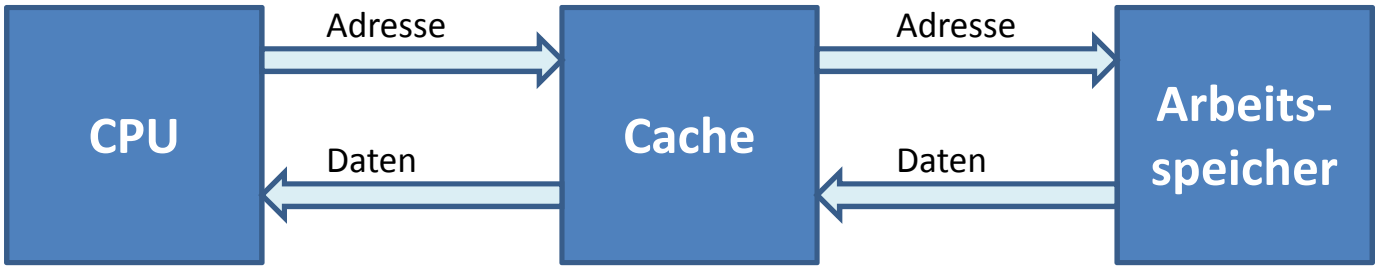
01010010 01001101 01000001 01010100, Paritätsbit: 1

Ja! Leider erlaubt das Paritätsbit nur Fehlererkennung und keine Fehlerkorrektur.
Das Paritätsbit gibt also keine Auskunft darüber welches Bit falsch übertragen wurde.

Caches

Was ist ein Cache ?

Caches sind Pufferspeicher, die Kopien von Inhalten eines anderen Hintergrund-Speichers beinhaltet. Bei CPUs kann der Cache direkt im Prozessor integriert oder extern auf der Hauptplatine platziert sein.



Vorteile?

- Verringerung der Zugriffszeit (weniger Zugriffe auf Hintergrundspeicher)
- Dadurch weniger Bandbreitenanforderung auf Hintergrundspeicher

Nachteile?

- Teuer (da schnellere Speichertechnologie , z.B. SRAM gegenüber DRAM)
- Dadurch sind Caches relativ klein und können nur wenige Daten speichern

Caches (2)

Caches nutzen aus …

1) Zeitliche Lokalität:

„Wenn Speicherzelle X jetzt gebraucht wird, wird X vermutlich gleich nochmal gebraucht.“

z.B. in Schleifen

```
Zahl = 52643.34
x = 0
Solange (Bedingung gilt)
        x = x + zahl * 5
        ……
```

⟹ Auf „Zahl“ wird in jeder Schleife zugegriffen, also ist es günstig diesen Wert im Cache zu halten.
(Erst wenn ein Wert lange nicht mehr benötigt wurde, ist es wahrscheinlich dass er gar nicht mehr gebraucht wird – dann kann er aus dem Cache geworfen werden um Platz für einen neuen Wert zu machen)

Caches (3)

Caches nutzen aus …

2) Räumliche Lokalität:

„Wenn Speicherzelle X gebraucht wird, braucht man vermutlich auch Speicherzelle Y in der Nähe von X.“

z.B. liegen die Befehle im Speicher oft nacheinander

z.B. in Schleifen:

```
i = 0
summe = 0
Solange (Bedingung gilt)
        summe = summe + Array[i]
        i = i + 1
```

⟹ Die Werte des „Arrays“ liegen hintereinander im Speicher und werden nacheinander benötigt. Somit ist es sinnvoll beim Lesen von Array[0] auch schon weitere Werte des Arrays in den Cache zu laden.

EBNF & Syntaxdiagramm

EBNF (Erweiterte Backus-Naur-Form)

Die Syntax einer Sprache legt mittels Regeln fest, welche Sätze zu der Sprache gehören.

- Syntax = formale Festlegung, wie der Programmcode aufgebaut sein muss (also z.B. vordefinierte Befehle, Aufbau von Schleifen u. Funktionen, …)

= wie Anweisungen geschrieben und verschachtelt werden können

Eine Grammatik ist eine Menge von Syntaxregeln, durch die ein Sprachschatz beschrieben wird.

⟹ Erweiterte Backus-Naur-Form(EBNF) ist eine weit verbreitete Schreibweise für Grammatiken.

EBNF - Metazeichen

Metazeichen dienen zur Beschreibung der Grammatikregeln, durch die die zu einer Sprache gehörenden Sätze festgelegt werden

Aufbau der EBNF:

=	trennt linke und rechte Regelseite
.	schließt Regel ab
|	trennt Alternativen
Beispiel: x | y	beschreibt: x, y
()	klammert Alternativen
Beispiel: (x | y) z	beschreibt: xz, yz
[]	wahlweises Vorkommen
Beispiel: [x] y	beschreibt: xy, y
{}	kein- bis n-maliges Vorkommen
Beispiel: {x} y	beschreibt: y, xy, xxy, xxxxy, ...

EBNF

Es wird zwischen Terminal- und Nichtterminalzeichen unterschieden.

Terminalzeichen: “a“... “z“ ⇐ In Anführungszeichen!
Nichtterminalzeichen: Zahl, Ziffer

Einfaches Beispiel: Adresse

Adresse = Straße Hausnummer.
Straße = Buchstabe {Buchstabe}.
Hausnummer = Ziffer {Ziffer}.
Buchstabe = “a“ | “b“ | “c“ | “d“ | … . ⇐ Korrekterweise müssen alle Terminalsymbole angegeben werden!
Ziffer = “0“ | “1“ | “2“ | “3“ | … .

EBNF (Aufgabe)

Adresse = Straße Hausnummer.
Straße = Buchstabe {Buchstabe}.
Hausnummer = Ziffer {Ziffer}.
Buchstabe = “a“ | “b“ | “c“ | “d“ | … .
Ziffer = “0“ | “1“ | “2“ | “3“ | … .

Lässt sich „weg3“ ableiten?

Adresse
-> Straße Hausnummer
-> Buchstabe {Buchstabe} Hausnummer
-> Buchstabe Buchstabe Buchstabe Hausnummer
-> Buchstabe Buchstabe Buchstabe Ziffer {Ziffer}
-> Buchstabe Buchstabe Buchstabe Ziffer
-> weg3

EBNF (Aufgabe)

Drücke folgende Bedingungen in Form einer EBNF aus!

Beliebiges Vorkommen von a gefolgt von einem b

⇨ {a} b

Beliebiges häufiges Vorkommen von a und b beginnend mit ab

⇨ ab{a|b}

Beliebige Zeichenkette aus a,b,c welche min. einmal “abc“ enthält

⇨ {a|b|c}abc{a|b|c}

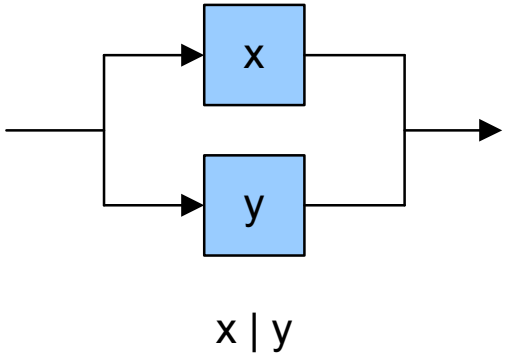
Zeichenkette aus a und b welche nicht mehr als zweimal “a“ hinter-einander enthält

⇨ {[a|aa]b}

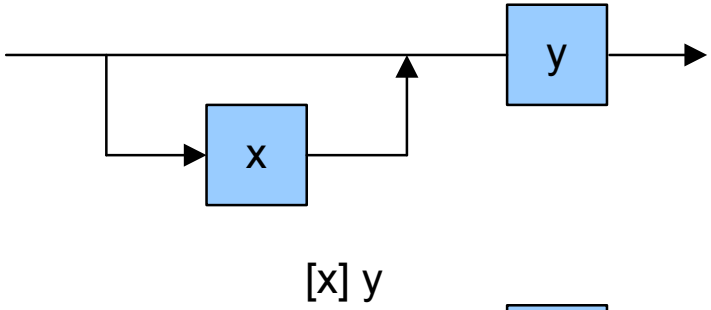
Syntaxdiagramme

- Syntaxdiagramme lassen sich in eine EBNF überführen bzw. können eine EBNF darstellen.
- Also eine grafische Darstellung der EBNF.

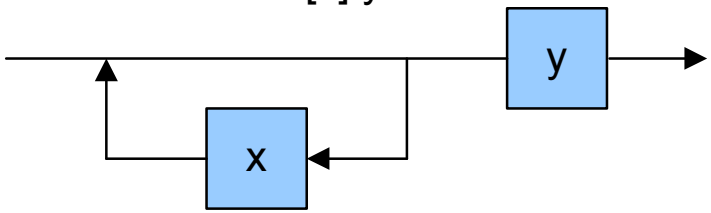
- Jedes EBNF-Metazeichen wird hierzu in Form einer graphischen Darstellung umgesetzt:



x | y



[x] y

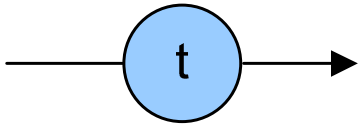


{x} y

Syntaxdiagramme (2)

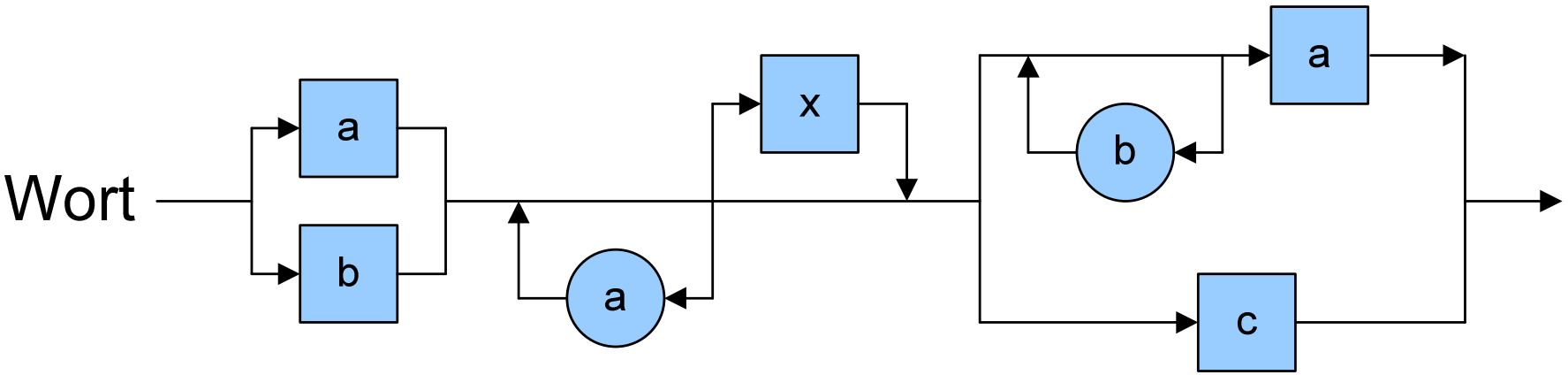
Nichtterminalsymbole im Syntaxdiagramm:

Terminalsymbole im Syntaxdiagramm:



Syntaxdiagramm (Beispiel)

Drücke folgendes Syntaxdiagramm in einer EBNF aus!



Lösung:

Wort = (a | b) {"a"} [x] ({"b"} a | c) .

EBNF – Aufgabe Dateiname

Es soll die EBNF zum Syntax eines Dateinamen angegeben werden, der nur aus Buchstaben und Ziffern besteht. Er soll mind. 1 Zeichen im Namen tragen und genau 3 Zeichen in der Erweiterung.!
(Die Terminale Buchstabe und Ziffer seien bekannt.)

- y.yxz
- dummy2.pgh
- 12345.123
- Xy1.1yX

Erlaubt!

- .gfd
- aldjf.a
- zu.laut
- haben.n?h

Verboten!

Lösung: Dateiname = (Buchstabe|Ziffer) {Buchstabe|Ziffer}'.'
(Buchstabe|Ziffer)(Buchstabe|Ziffer)(Buchstabe|Ziffer).

Java / Rechnerübungen

- Ab nächste Woche Rechnerübungen! Bitte per E-Mail anmelden.
- Praxisaufgaben bis spätestens 3 Wochen nach Abgabe vorführen!
- Java-Installationsanleitung auf der Informatik 1 Homepage verfügbar
- In den Poolräumen ist Java bereits installiert. Wer einen Laptop hat, kann seine Aufgaben auch darauf vorführen.
- In/Out Klassen von der Info1 Homepage verwenden. Wichtig: Ins gleiche Verzeichnis kopieren wie der QuellCode.
- Wichtig: Name der Quellcodedatei muss mit Klasse übereinstimmen.
- Für Übungsblatt 3: Quellcode ausdrucken und mitabgeben.
  Später: Lösungen per E-Mail schicken ist in Ordnung.
- Die Lösung, die ihr abgibt, muss auch vorgeführt werden.

Das wars für heute…

Fragen ???

Viel Spaß mit dem Übungsblatt!