

Projekt: Entwicklung einer Infsii-Turtle-Code-Programmierungsumgebung

Entwicklung eines Interpreters

Wir wollen in Anlehnung an Eckart Modrows¹ Projektvorschlag „LOGO für Arme“ in Snap! (oder einer anderen aus dem Unterricht bekannten Programmierungsumgebung) eine textbasierte Bildbeschreibungssprache ähnlich zur Programmiersprache Logo entwickeln. Wir nennen diese selbst entwickelte Sprache `Infsii-Turtle-Code`.

Im Folgenden wird davon ausgegangen, dass das eigentliche Projekt sowie die einzelnen Bestandteile bereits in der Lerngruppe besprochen wurden (vgl. die zugehörigen Infsii-Materialien). Wir konzentrieren uns an dieser Stelle auf die schrittweise Entwicklung eines Interpreters. Dabei wird davon ausgegangen, dass eine andere Gruppe die Implementierung eines zugehörigen Parsers vornimmt.

Wir entwickeln unsere `Infsii-Turtle-Code-Programmierungsumgebung` mithilfe von Snap! oder einer anderen aus dem Unterricht bekannten Programmiersprache. Das bedeutet, dass unser Turtle-Code-Interpreter den Infsii-Turtle-Code in diese Programmiersprache übersetzen muss, der darin enthaltene Interpreter übernimmt dann die weitere Ausführung. Dabei kann der Turtle-Code-Interpreter davon ausgehen, dass der Parser syntaktisch falsche Eingaben erkannt hat und nur zur Sprache gehörende Ausdrücke weitergegeben wurden. Wenn Sie sich auf eine konkrete Sprachdefinition geeinigt haben und von syntaktisch korrekten Ausdrücken ausgehen, können Sie eigenständig mit der Entwicklung eines Interpreters beginnen. Unter Umständen ist dabei eine Modellierung mit einem geeigneten Automatenmodell (z.B. einem Mealy-Automaten) hilfreich. Alternativ können Sie sich an den folgenden Beispielen orientieren und diese jederzeit an Ihre eigene Sprachdefinition anpassen und gegebenenfalls erweitern

Sprachdefinition:

Wir gehen im Folgenden von folgender Sprachdefinition aus:

Befehl	Bedeutung
F (x)	Die Turtle bewegt sich um x Schritte nach vorne (F orward).
U	Der Stift wird deaktiviert (U p).
D	Der Stift wird aktiviert (D own).
T (x)	Die Turtle dreht sich um den Winkel x im Uhrzeigersinn nach rechts (T urn).

Tabelle 1: Erste Version einer Turtle-Sprache

¹ vgl. Modrow, Eckart, Theoretische Informatik mit Delphi, emu-online, 2005 bzw. Modrow, Eckart, „Einführung in die Informatik – Teil VII Erkennende Automaten“, vlin-Material

Aufgaben:

- Geben Sie an, welche Ausgabe durch die Befehlskette `DF (20) UF (20) DT (90) F (22)` erzeugt wird.
- Geben Sie eine Befehlskette zur Erzeugung eines gleichseitigen Dreiecks an.
- Übersetzen Sie die Beispielfehlskette aus Aufgabenteil a) in ein Snap!-Programm oder eine andere aus dem Unterricht bekannte Programmiersprache².
- In Abbildung 1 ist ein (vereinfachter) Übergangsgraph eines Mealy-Automaten zur Modellierung des Interpreters angegeben. Dabei steht `z` für eine beliebige Ziffer und `zahl` für eine Variable. Außerdem wurden die stets aufeinanderfolgenden Eingabezeichen `F` und `(` bzw. `T` und `(` zusammengefasst. Interpretieren Sie den Graphen. Erläutern Sie dabei insbesondere die Bedeutung der Variablen `zahl`.

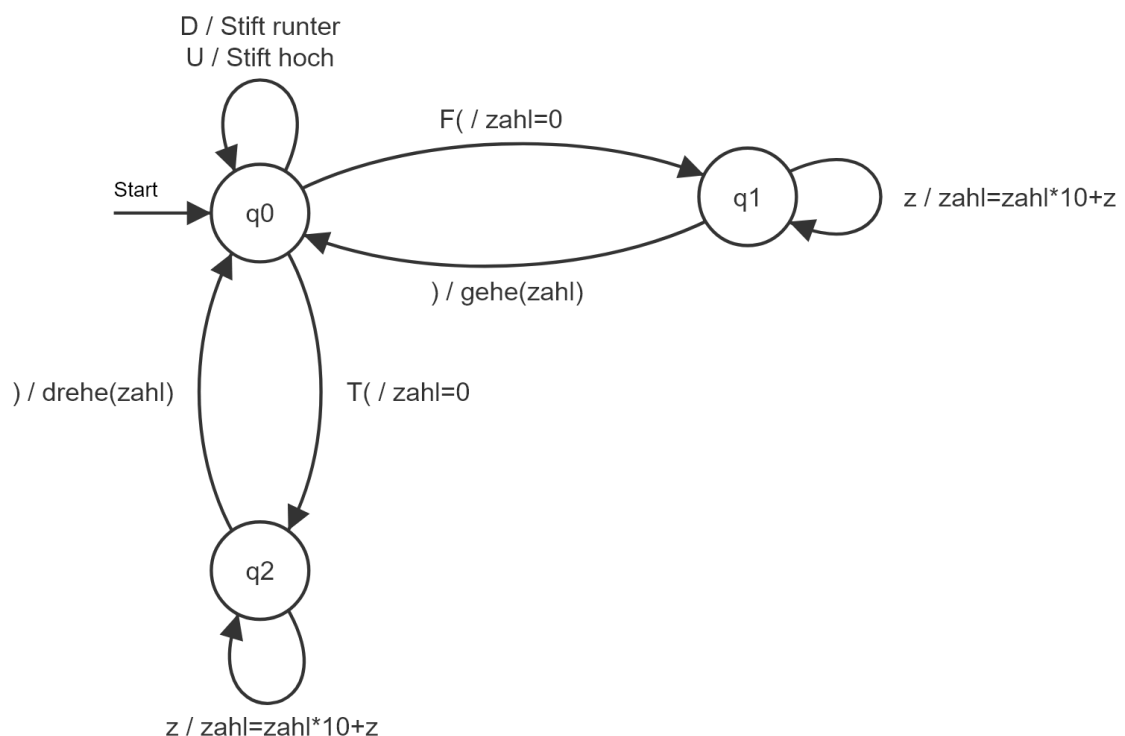


Abbildung 1: Übergangsgraph eines Mealy-Automaten zur Modellierung eines Interpreters

- Implementieren Sie ausgehend vom Übergangsgraphen in Abbildung 1 einen Interpreter. Sie können sich dabei am „rezeptartigen Vorgehen“ aus dem Material „Implementierung_Automat“ orientieren:

- Die einzelnen Zustände werden durch eine ganzzahlige Variable `zustand` kodiert. Diese hat zu Beginn den Wert `zustand=0`.
- Die Eingabe wird mithilfe einer Schleife vollständig durchlaufen.
- Abhängig vom aktuellen Zustand und dem aktuellen Eingabezeichen wird mittels einer geschachtelten Verzweigung der Folgezustand berechnet. Außerdem erfolgt möglicherweise eine Ausgabe/Aktion.

² Viele Programmiersprachen bieten bereits vorgefertigte Turtle-Klassen an, deren Befehle hier genutzt werden können. Alternativ kann ein Teil der Lerngruppe die Implementierung einer eigenen Turtle-Klasse übernehmen.

Mögliche Erweiterungen der Sprachdefinition

Die Infsii-Turtle hat zunächst nur wenige Befehle. In der folgenden Tabelle werden weitere mögliche Befehle vorgestellt. Diese Liste ist keineswegs vollständig und kann um eigene Ideen ergänzt werden.

Aufgabe: Wählen Sie mehrere neue Befehle aus und überlegen sich geeignete Codierungen. Erweitern Sie anschließend Ihr Automatenmodell des zugehörigen Interpreters entsprechend und implementieren Sie es.

Befehl	Bedeutung
	Die Turtle bewegt sich um x Schritte rückwärts.
	Die Turtle dreht sich um den Winkel x entgegen dem Uhrzeigersinn.
	Die Stiftstärke wird geändert.
	Die Stiftfarbe wird geändert.
	Die Stiftfarbe wird auf den RGB-Wert (r, g, b) gesetzt.
	Die Turtle versteckt sich.
	Die Turtle wird sichtbar.
	Der Bildschirm wird gelöscht.
	Die Turtle geht zur Position (x, y) .
	Eine Befehlsfolge wird x-mal wiederholt.

Dieses Werk ist lizenziert unter einer [Creative Commons Namensnennung - Nicht-kommerziell - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz](https://creativecommons.org/licenses/by-nc-sa/4.0/). Sie erlaubt Bearbeitungen und Weiterverteilung des Werks unter Nennung meines Namens und unter gleichen Bedingungen, jedoch keinerlei kommerzielle Nutzung.

Haftung Quellcode:

Für die korrekte Ausführbarkeit der Quelltexte in diesem Material wird keine Garantie übernommen. Auch für Folgeschäden, die sich aus der Anwendung der Quelltexte oder durch eventuelle fehlerhafte Angaben ergeben, wird keine Haftung oder juristische Verantwortung übernommen.

Bildnachweis: Die Abbildungen zu endlichen Automaten wurden mit Flaci erstellt:

<https://flaci.com/home/> (Link vom 14.05.2020).