Superturingmaschinen

Felix Karg

7. Dezember 2017

Abstract:
Superturingmaschinen sind eigentlich nur Turingmaschinen, die wir nach dem Ende der Zeit betrachten können, wodurch wir also z.B. das Klassische Halteproblem lösen. Turingmaschinen sind ein Klassisches Beispiel für Hypercomputation, also eine Berechenbarkeitsstufe über traditionellen Turingmaschinen, die allerdings Physisch fragwürdig realistisch umzusetzen ist. Dennoch ergeben sich ein paar interessante Eigenschaften, die genauer beleuchtet werden sollen.

1 Berechenbarkeit

Es gibt (wie allgemein bekannt ist) verschiedene Stufen der Berechenbarkeit, wobei die einfachste Kombinatorische Logik (einfache, nicht-Zyklische Schaltkreise) ist, und weitergehende über DEA (Determistische Endliche Automaten), PDA (Push-Down-Automaton, dt: Kellerautomaten) bis hin zu Turingmaschinen, die schon bereits alles Berechenbare berechnen können. Häufig wird also z.B. eine Programmiersprache daran gemessen, ob sie Turingvollständig ist, ob also alle Berechenbaren Funktionen berechnet werden können. Neben Turingmaschinen gibt es (in den für realistische berechenbarkeit betrachteten Bereichen) auch andere Modelle der Berechenbarkeit, z.B. Registermaschinen, Generell-Rekursive Funktionen oder das Lambda-Kalkül.

1.1 Konstrukt Turingmaschine

Eine Turingmaschine ist eigentlich eine einfache, konzeptionell aber robuste Konstruktion. Ich werde gleich noch näher darauf eingehen dass es verschiedene, aber im Endeffekt äquivalente, Turingmaschinen gibt. (hier keine formale Definition, da es nicht Hauptteil des Vortrages ist sowie wenig hilfreich für das Verständnis).
Im Kern besteht eine Turingmaschine aus folgenden Elementen: Ein Band, das ein Anfang hat aber nach rechts hin unendlich lange ist, also kein Ende hat. Die eigentliche Maschine besteht aus einem kombinierten Lese- und Schreibkopf, welcher auf dem Band operiert. Neben dem Initialzustand hat sie lediglich eine endliche Menge an Zuständen.

Pro 'Berechnungsschritt' passiert nun folgendes: Die Turingmaschine liest ein Zeichen vom Band und entscheidet in Abhängigkeit ihres momentanen Zustandes was sie als nächstes tut. Möglich sind: Das aktuelle Zeichen so belassen, es überschreiben, das Band nach rechts oder links bewegen, sowie entsprechend der Übergangsfunktion eventuell den Zustand wechseln.

1.2 Eigenschafen von Turingmaschinen

Relevant für uns ist im Folgenden, dass es gewisse Definitionen beziehungsweise Arten von Turingmaschinen gibt, die der vorgestellten Äquivalent sind. Ein Beispiel wäre eine Turingmaschine die auf Zwei oder mehreren Bändern rechnet (die beide Unendlich lang sind), eine die auf einem Band rechnet das in beide Seiten unendlich lange ist, eine die in ihrem Alphabet nur $\{0, 1\}$ hat sowie eine die ein Beliebig langes, endliches Alphabet zur verfügung hat. Eine sehr interessante Eigenschaft ist, dass unsere Turingmaschine mit ihren Endlichen vielen Zuständen eindeutig definiert ist, Sie also auch entpsrechend Codiert werden kann und von einer anderen gelesen und Simuliert werden kann. Daraus ergibt sich auch dass eine Turingmaschine gleichzeitig mehrere simulieren kann, wofür man nun einfacherhalber gerne eine mit mehreren Bändern verwendet, um die Simulierten Turingmaschinen seperat zu halten und genauer betrachten zu können.
Außerdem gibt es das Halteproblem, also dass bereits bei einer Turingmaschine mit sehr wenigen Zuständen kein Zuverlässiger Algorithmus existieren kann, der entscheidet ob diese irgendwann hält (Terminiert, ein Ergebnis liefert) oder eben nicht.

1.3 Aussagentypen

Eine Teilmenge $M \subseteq \mathbb{N}$ ist genau dann von einer Turingmaschine aufzählbar (oder Rekursiv aufzählbar) sofern es eine Turingmaschine gibt, die in beliebiger Rheinfolge alle Elemente dieser Menge auf das Band schreiben kann. Eine solche Menge wird Σ_1-Menge genannt, wenn es eine Σ_1-Aussage ϕ gibt, die über Logik (Arithmetik) erster Stufe aufgebaut ist, also der form $\phi = \exists m_1 ... \exists m_k \heartsuit$, wobei die Quantoren über natürliche Zahlen reichen dürfen sowie in $\heartsuit$ keine weiteren ungebundenen Quantoren vorkommen dürfen und nur eine Freie Variable darin vorkommen darf, sodass $M = \{n \in \mathbb{N} \mid \phi(n)\}$.

1.4 Limitierungen von Turingmaschinen

Normale Turingmaschinen sind nun offensichtlich nicht in der Lage, Π_1-Aussagen zu entscheiden, was also nun Formeln statt mit $\exists$-Quantoren mit $\forall$-Quantoren sind (eine solche Turingmaschine würde aus offensichtlichen Gründen nicht halten). Außerdem gibt es das bereits angesprochene Halteproblem.

2 Unendlichkeit

Umgangssprachlich verwendet man einfach das Wort 'Unendlich'. Da damit zwar meist klar ist was gemeint ist, dies aber nicht immer das selbe ist möchte ich zumindest die folgenden beiden Arten der Unendlichkeit differenzieren, von denen eine gemeint ist, und die andere explizit nicht. Ich beschreibe beide nun als jeweils neue Klassen von Zahlen, die allerdings Konzeptionell nichts neues, sondern altvertrautes sein sollten.

2.1 Kardinalzahlen

Kardinalzahlen sind Zahlen die Kardinalitäten angeben, also Beispielsweise die Anzahl von Elementen in einer Menge ($|\{\square, \nabla, \heartsuit\}| = 3$). Soweit sind Kardinalzahlen also nicht von Natürlichen Zahlen zu unterscheiden, aber wir sind ja auch noch bei endlichen Mengen. Sobald wir unendliche Mengen betrachten, Beispielsweise $\mathbb{N}$, wird es interessant. $|\mathbb{N}| = \aleph_0$, $\aleph$ ist nun ein Buchstabe aus dem Häbräischen Alphabet. Wie bekannt sein sollte ist $\mathbb{N}$ abzählbar unendlich Groß, genauso wie $\mathbb{Z}$ und $\mathbb{Q}$. Es ist also klar, dass $\aleph_0 + \aleph_0 = \aleph_0$ ($\mathbb{Z}$), sowie $\aleph_0 * \aleph_0 = \aleph_0$ ($\mathbb{Q}$). Dies ist eine Art der Unendlichkeit auf die ich im Folgenden nicht weiter eingehen werde, wer ein Tieferes verständnis diesbezüglich michte, ein schönes Gedankenexperiment dazu ist Hilberts hotel. Ich werde hier nicht weiter auf z.B. die Reellen Zahlen eingehen, da nicht feststeht welche Kardinalzahl diese Haben, bis auf dass diese größer ist als $\aleph_0$.

2.2 Ordinalzahlen

Nun zu einer anderen Art von Unendlichkeit, mit der wir auch vertraut sind. Wir fordern eigentlich nur eine Eigenschaft: und zwar dass die Ordnung erhalten bleibt. Anfänglich sind wir wieder äquivalent zu den Natürlichen Zahlen, da diese bereits eine wohlordnung haben. Wenn wir nun einen Zahlenstrahl betrachten, Zeichnen wir einen solchen häufig mit ∞, umgangssprachlich unendlich. Dieses Zeichen ist nicht Element der Natürlichen oder sogar Reellen Zahlen, sondern bezeichnet eher ein Element größer als alle die wir aufschreiben oder uns vorstellen könnten. Wir definieren uns nun ein Element ω, das direkt rechts daneben anzuordnen ist, also das erste Element außerhalb, oder neben, eher nach, dem Zahlenstrahl. Es sollte klar sein dass es strikt größer ist

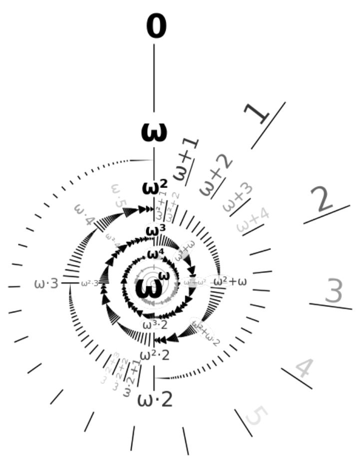

Abbildung 1: Ordinaler Zahlenstrahl

als alle vorhergehenden Ordinalzahlen, also z.B. alle Zahlen die auch Natürliche Zahlen wären. Die frage nach $\omega - 1$, also der Ordinalzahl vor ω, kann nicht beantwortet werden. Intuitiv: es ist nicht möglich eine Feste Zahl direkt vor 'unendlich' zu definieren, denn könnten wir das, könnten wir diese Zahl so weit erhöhen (um 2) und wären größer als ω, die ja die größte Zahl sein soll. Allerdings ist zu beachten, dass $1 + \omega = \omega \neq \omega + 1$. Intuitiv kann man wieder sagen dass eben kein Element davor, wohl aber ein strikt größeres Element danach gibt, oder dass man einen Zahlenstrahl wohl auch bei 2 anfangen kann, dieser entsprechend immernoch gleich lang ist, aber wenn man ein Element nach dem Zahlenstrahl hinzufügt, muss es immer erst nach dem Zahlenstrahl, strikt hinter allen anderen Elementen davor, angeordnet sein. Genauso ist $\omega + \omega = \omega * 2$, oder $\omega * \omega = \omega^2$.

3 Superturingmaschinen

Superturingmaschinen klingen jetzt erstmal total super, und das sind sie auch. Eigentlich sind es nur Turingmaschinen, außer dass ihre Schritte in Ordinalzahlen gezählt werden, dass es also Schritte ω, $\omega + 1$, etc. quasi nach dem Ende der Zeit gibt, wo sie einfach weiterrechnen können. Dadurch lösen wir das klassische Halteproblem von Turingmaschinen, da wir einfach schauen können ob unsere Superturingmaschine (die sich ja sonst nicht unterscheidet) nach dem Ende der Zeit bereits angehalten hat oder eben nicht.

3.1 Verhalten bei Grenzen

Was natürlich immer passieren ann, ist dass eine Superturingmaschine nicht anhält, und ständig weiter eine Zelle mit 1 und anschließend mit 0 beschreibt. In einem solchen fall (oder allgemein, wenn sie nicht hält) muss definiert werden, was anschließend, in schritt ω die Zustand sein soll. Meta: An diesem Punkt kann auf Supertasks hingewiesen werden, ein Beispiel wäre das umschalten eines Lichtschalters nach immer halben Zeitintervall zuvor, also angefangen bei 1s, 0.5s, 0.25s, ... wäre er in Sekunde 2 bei ω angelangt, allerdings ist bei ständigem umschalten eines Lichtschalters nicht definiert, in welchem Zustand er anschließend, also nach sekunde 2, ist. Außerdem gibt es auch keine eindeutige Antwort, da es gleichzusetzen wäre mit 'Ja, Unendlich ist gerade' oder eben dem Gegenteil davon. Allerdings sollte das Publikum bereits mit Supertasks im allgemein vertraut sein. Da davon nicht auszugehen ist, werde ich sie kurz erwähnen, aber dann weitermachen mit: eine Turingmaschine die nicht Terminiert, in welchem Zustand ist sie zum Zeitpunkt ω? Weil: es kann einer von vielen sein, gleichzeitig eines von auf die verschiedensten weisen beschriebenes Band. Und wieder: egal wie man's festlegt, kann es gut passieren dass Eindeutig ist wie viele und welche Schritte bisher notwendig gewesen sein müssen, also dass ω in einem solchen fall nicht 'nach' dem Zahlenstrahl kommt. Dementsprechend definiert man es folgendermaßen: Sofern die Turingmaschine hält, ist sie auch nach beliebig vielen weiteren schritten in ihrem Finalen zustand. Ist dies allerdings nicht passiert, wird der Schreib-/Lesekopf wieder auf den Anfang gesetzt, sowie die Superturingmaschine auf ihren Startzustand. Der einfachheit nehmen wir an dass wir nur nuller (default) und einser auf unserem Band haben können. Eine null wird nun an Schritt ω überall dort stehen wo entweder keine Eins geschrieben wurde, oder nur endlich oft eine Eins geschrieben wurde. Eine Eins steht hingegen überall dort, wo zu mehr als endlich vielen Zeitpunkten eine Eins stand, bis hin zu von anfang an (durchgehend). Dementsprechend ist der Wert einer jeden Zelle entsprechend ihres Limeswertes, und die Turingmaschine ist wieder auf Anfangszustand, allerdings mit möglicherweise verändertem Band.

3.2 Fähigkeiten

Superturingmaschinen können nun Natürlich zum einen alles tun was normale Turingmaschinen auch können. Außerdem können sie, wie wir eben gesehen haben, entscheiden ob eine normale Turingmaschine hält oder nicht. Genauso können sie natürlich neben dem Simulieren von normalen Turingmaschinen auch Superturingmaschinen simulieren. Was sie allerdings auch entscheiden können sind Σ^1_1-Aussagen, also Aussagen der Form 'Es gibt eine Funktion $\mathbb{N} \to \mathbb{N}$ so dass ...', sowie Π^1_1-Aussagen, also Aussagen der Form 'Für jede Funktion $\mathbb{N} \to \mathbb{N}$ gilt ...'.

3.3 Halteverhalten von STM