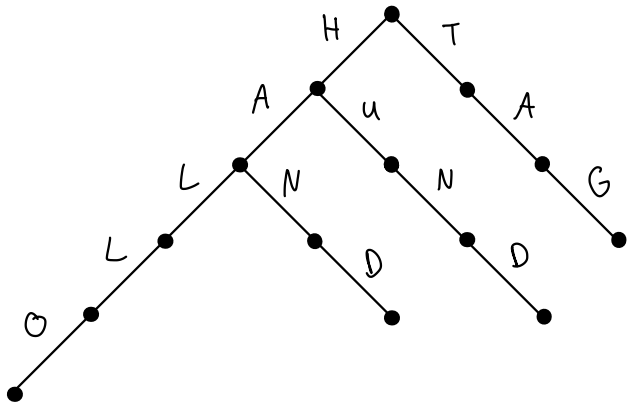
ALP III - VL 21

Wdh.

- Stringsuche: Rabin-Karp Algorithmus
  - Beschleunigt den naiven Algorithmus, indem es eine Hashfunktion verwendet, um festzustellen, ob es wahrscheinlich ist, dass das Muster an einer Stelle vorkommt
  - Die spezielle Struktur erlaubt es, die Hashfunktion für alle Stellen in s in $O(k)$ Zeit auszurechnen
  - Fingerprinting: Stelle eine Zeichenkette kurz durch ein Hashwert dar. Hat viele weitere Anwendungen.

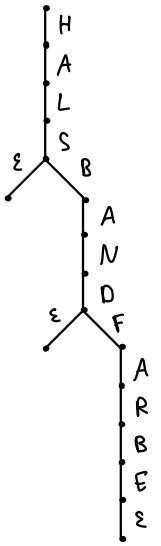
Tries:
- Implementierung der ADT geordnetes Wörterbuch wenn es sich bei den Schlüsseln um Zeichenketten handelt
- Gewurzelter Mehrwegsuchbaum, in dem jeder innere Knoten $1, \ldots, |\Sigma|$ Kinder hat
- Jeder Kinderknoten eines inneren Knoten ist mit einem Zeichen $\sigma \in \Sigma$ beschriftet, so dass jedes Zeichen höchstens einmal vorkommt
- Die Blätter entsprechen den Zeichenketten im Trie



Problem: Was, wenn ein Wort Präfix eines anderen ist?

- Einfache Lösung: Sei $\varepsilon \notin \Sigma$ ein neues Zeichen,
  füge ε an jede Zeichenkette hinten an

$S = \{HALS, HALSBAND, HALSBANDFARBE\}$



Speicherbedarf: $\mathcal{O}(\sum_{s \in S} |s|)$

Die Operationen put, get, pred, succ können in $\mathcal{O}(k|\Sigma|)$ implementiert werden,
(pred, succ: lexicographisch)

wobei k die Gesamtlänge der beteiligten Strings ist.

(d.h. put(s), get(s) } braucht $\mathcal{O}(|s||\Sigma|)$ Zeit

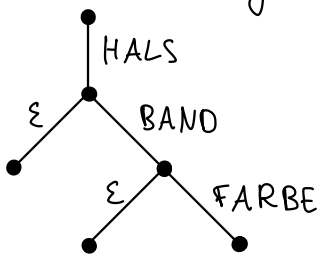
pred(s), succ(s) } braucht $\mathcal{O}((|s|+|s'|)|\Sigma|)$ Zeit, wobei s' der Vorgänger/Nachfolger von s ist)

Komprimierte Tries / Patricia Tries

1968 Donald R. Morrison

P	ractical
A	lgorithm
T	o
R	etrieve
I	nformation
C	oded
I	n
A	lphanumeric

Idee: Pfade, die aus Knoten mit jeweils nur einem Kind bestehen, zu einer Kante zusammenführen, die mit dem Teilstring beschriftet ist



Heißen auch PATRICIA-Tries

→ Nur $O(n)$ statt $O(\sum_{s \in S} |s|)$ Knoten, da ≤ 2 Knoten pro Wort benötigt werden

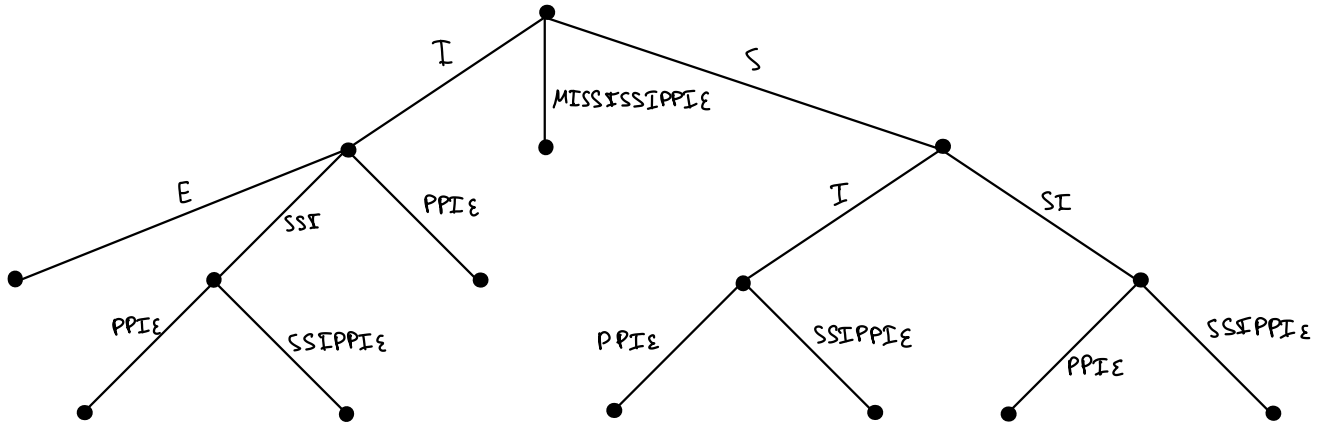
Aber: Gesamtgröße der Labels ist immernoch $O(\sum_{s \in S} |s|)$ (static)

Suffixe-Bäume:

Ein Suffixe-Baum ist ein komprimierter Trie, der alle Suffixes eines Strings speichert

MISSISSIPPIε

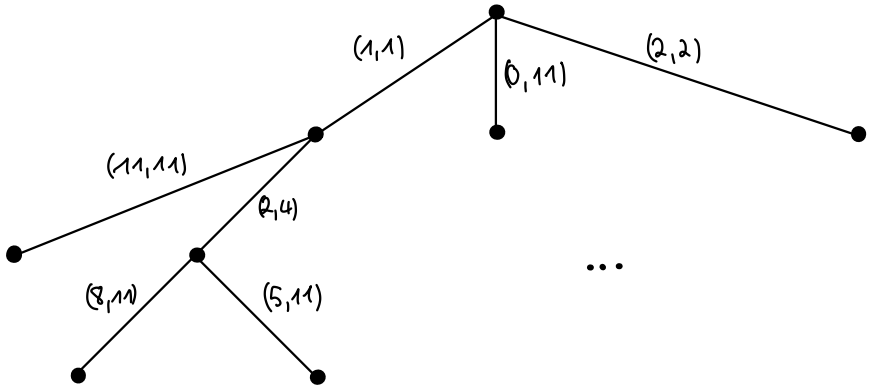
Suffixes: MISSISSIPPIε, ISSISSIPPIε, SSISSIPPIε, SISSIPPIε, ..., ε



Der Suffixbaum hat $O(n)$ Knoten, aber die Gesamtlänge der Beschriftungen kann $\Theta(n^2)$ sein.

Lösung: Statt des Substrings, speichere Zeiger in den String

0	1	2	3	4	5	6	7	8	9	10	11
M	I	S	S	I	S	S	I	P	P	I	ε



Dann benötigt der Suffixbaum nur $O(n)$ Platz

Anwendung: Verarbeite einen langen Text vor, so dass man nach vielen verschiedenen Mustern effizient suchen kann

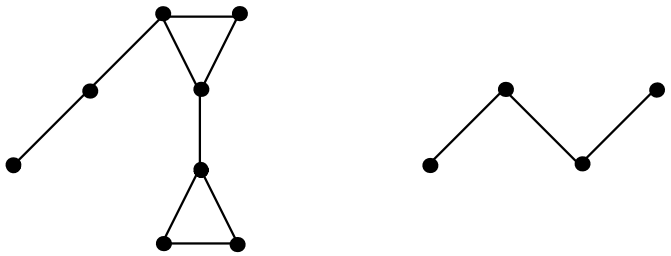
Wenn man den Suffixbaum für S hat, kann man in $O(|t||\Sigma|)$ Zeit entscheiden, ob t in s vorkommt

Bemerkung: Man kann einen Suffixbaum zu einem Wort der Länge n in $O(n)$ Zeit konstruieren, durch einen speziellen Algorithmus

Graphen

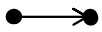
Ein Graph $G=(V,E)$ besteht aus einer Menge V von Knoten und eine Menge E von Kanten.

Die Kanten sind Teilmengen der Größe 2 der Knotenmenge.



Varianten: • Multigraph: erlaube Doppelkanten und Schleifen

• gerichteter Graph: Kanten sind geordnete Paare



• gewichteter Graph: Kanten ist eine Zahl aus $\mathbb{R}$ zugeordnet

Bsp für Graphen:

- Eisenbahn
- Facebook
- Stromnetz
- Autobahn
- Beziehungen zwischen Proteinen