Es geht um Den Besitz und die Herrschaft von geteilten digitalen Inhalten.

Einleitung

In dieser Arbeit möchte ich eine Meta-Programmiersprache ($SCFG$) für eine beliebige eindeutige kontextfreie Grammatik G herrausarbeiten, die das verteilte Programmieren ermöglichen soll. Hierfür werden in (1) zwei Funktion angegeben. Eine erweitert eine gegebene kontextfreie Grammatik zu einer $SCFG$, die andere erzeugt aus einem Wort der Sprache $L(SCFG)$, ein Wort in der ursprünglichen Sprache $L(G)$. Manipulationen eines Wortes einer SCFG-Sprache sind bedingungen unterworfen, die in (2) angegeben sind.

1. Meta - Programmiersprache

Einleitung

Die Idee hinter der erweiterung der SCFG Sprache ist, dass der Besitz eines Programms eindeutig beteiligten **Akteuren** zugeordnet ist. Die Zuordnung repräsentiert auch das mitspracherecht der Akteure. Manipulationen am Programm können jederzeit vorgenommen werden, solange die Manipulation ein valides Wort in $L(G)$ ergibt. Manipulierte Wörter kommen als Option zu den bestehenden Wörtern hinzu, sodass Informationen nicht verlohren gehen. Optionen können durch Akteure bewertet werden.

Eine Konsensfunktion bestimmt dann die Option, die die beste Bewertung durch die Mehrheit der Akteure erhalten hat. Sie gilt als **gerecht**, wenn sie die besitzgewichtung der Akteure berücksichtigt.

Sprach-Erweiterung

Abhängig von einer gegebenen Programmiersprache $G = (N, T, S, P)$ soll eine Erweiterte Programmiersprache $L(SCFG)$ konstruiert werden. Die Wörter der neuen Sprache repräsentieren ein geteiltes Programm O und besitzen folgende Informationen:

1. $n \in \mathbb{N}$ Akteure
2. eindeutige **Besitzgewichtung** von Akteuren w_i für den Akteur $0 \leq i \leq n$
3. eine Menge von validen Programmen mit mindestes einem Element
$$P_O \subseteq \{w | S \rightarrow_P^* w \wedge w \in T^*\} \wedge |P_O| \geq 1$$
4. eine Bewertung der Programme durch die Akteure.
$$rate_O : A \times P \rightarrow [0, 1]$$

Zu dem wird eine Compilierfunktion gesucht, die Wörter der $L(SCFG)$ Sprache, wörter der Ursprünglichen Sprache $L(G)$ zuweist.

$$consens_O : L(SCFG) \to L(G)$$

Qualitätskritärien

1. die wörter der Sprache $L(SCFG)$ sind möglichst klein
2. es ist einfach Optionen zu Bewerten: Die Anzahl der transitionen ist möglichst klein, um auf ein gewünschtes Bewertungsprofil zu kommen.

   Sei $a \in A$ ein Akteur, sei $X_i \subseteq P \times [0,1]$ die derzeitige Bewertung der Programme von a und sei $X_j \subseteq P \times [0,1]$ die gewünschte Bewertung. Sei t eine von a gültige Transformation, so dass $X_k \to_t X_{k+1}$ und sei T die Menge der von a gültigen Transformationen. Die Bewertung ist einfach, wenn für $X_i \to_T^n X_j$ n möglichst minimal ist.

Besitz Der Besitz eines Objekts O wird durch das Recht bestimmt, dieses Objekt kontrollieren zu können. Besitzen mehrere Akteure ein Objekt, so gibt der Anteil am Objekt an, zu welcher gewichtung jeder einzelne Akteur über das Objekt mit bestimmen kann.

Seien a **Akteure** und sei A die Menge aller Akteure.

Die Funktion $share_O : A \to \mathbb{N}$ gibt den Teil von a an, der im Besitz von einem Akteur $a \in A$ ist. Den Anteil am Objekt.

Gilt $share_O(a) > 0$, so nennt man a auch ein **Member** von O.

$M_O \subseteq A$ ist die Menge aller Member von O.

$|O| := \sum_{m \in M_O} share_O(m)$ gibt die **Größe** des jeweiligen Objekts an.

Bewertung

Diese werden entweder **direkt**, also vom Akteur selber, oder **indirekt** also durch eine Delegation an einen anderen Akteur angegeben.

1.1 erweiterung einer CFG zu einer SCFG

Sei $G = (N, T, S, P)$ eine eindeutige contextfreie Grammatik.

Eine Erweiterung auf eine SCFG definieren wir als $S(G) := (N', T', S', P')$, mit:

$N' := N \cup \{O, D, A\}\ mit\ O, D, A \notin N$

$T' := T \cup \{[,], \oplus, number, float, hash\}\ mit\ [,], \oplus \notin T$

number stellt eine ganzzahlige Nummer dar. **float** stellt eine fließkommazahl dar. **hash** ist ein identifikatior für einen akteur, hier ein 64 bit hex string.

$P' := P \cup P_{Options} \cup P_{Start} \cup P_{Delegations} \cup P_{Voting} \cup P_{Acteurs}$

$P_{Options} := \{R \to [O][D], O \to r \oplus [V]O | R \to r \in P \text{ mit } r \in \Sigma^*\} \cup \{O \to \varepsilon\}$

$P_{Start} := \{S' \to [A]S\}$

$P_{Delegations} := \{D \to [hash\ hash]; D, D \to [hash\ hash]\}$

$P_{Voting} := \{V \to [hash\ float]V, V \to \varepsilon\}$

$P_{Acteurs} := \{A \to [hash\ number]A, A \to \varepsilon\}$

zu zeigen

- eindeutigkeit der SCFG

1.2 erzeugung eines Wortes der Ursprungsgrammatik aus einem Wort der SCFG

Die Idee hier ist, dass die transitive Hülle der Delegationen als Attribut während der Ableitung vererbt(inherited) wird. Die Votemenge wird hingegen als Attribut synthetisiert. Die Delegationen zusammen mit den Stimmen quantifizieren jede Option aus einer Optionsmenge und wählen eine Konsens-Option.

- zirkuläre delegationen
- implekation von Arrow's Theorem auf strategisches voten

2. Manipulation der Wörder einer SCFG-Sprache

Die Idee hinter der manipulation ist des es sich bei den Wörter um eine Persistente Datenstruktur handelt. Demnach können Optionen nur hinzu kommen, jedoch nicht gelöscht werden. Jeder Akteur kann Delegationen und Stimmen abgeben sowie die eigenen löschen, hat jedoch keine Macht über die Stimmen und Delegationen der Anderen.

Die Manipulation wird durch ein **initialisiertes Transitionssysthem** T beschrieben.

Sei $G = (N, T, S, P)$ eine eindeutige Grammatik sowie $G' = S(G) = (N', T', S', P')$ die dazugehörige SCFG.

Qualitätskritärien

1. operationen wie $vote, delegate, addOption$ sind effizient in bezug auf Laufzeit und Speicherverbrauch.
2. Die Verifikation der Transaktion durch das Netzwerk soll möglichst effizient sein.

zu untersuchen

States

Ein Zustand der ist eine Struktur der form $(U, compile, w)$

U ist das Universum der Struktur und wird definiert durch:

$$U = \{w | S' \rightarrow_p^* w \wedge w \in T'^*\}$$

compile ist die in (1.2) definierte Funktion der Form: $compile : L(G') \rightarrow L(G)$

$w \in U$ ist das derzeitige Wort

Transitionen

Bei der Interaktion eines Programms, ist der Akteur bekannt und wird durch ein Public Key ausgewiesen. Der Akteur kann in einem Interaktionsschritt das derzeitige Wort altuallisieren, wenn es bestimmte Bedingungen erfüllt. Eine Interaktion ist also ein Tupel (h, w') so, dass h der Public Key des Akteurs ist, sowie w' das neue Wort.

Erzeugen einer Optionsmenge Sei $\alpha, \beta, o_1, o_2 \in T'^*$ sowie $R \in N$ und $V \rightarrow_P^* v$ Sei $w = \alpha o_1 \beta \oplus [v]$ Sei weiter $R \rightarrow_p^* o_1$ sowie $R \rightarrow_p^* o_2$ Dann ist $w := \alpha[o_1 \oplus [v] o_2 \oplus []][]\beta$ eine valide Transition.

Erweitern einer Optionsmenge Sei $w = \alpha[o]\beta \wedge S' \rightarrow_P^* \alpha R \beta$ mit $O \rightarrow_P^* o, o \in T'^*, R \in N$ Dann ist $w := \alpha[or \oplus []]\beta$ mit $R \rightarrow_P^* r, r \in T'^*$ eine valide Transition.

Hinzufügen einer Stimme Sei $w = \alpha r \oplus [v]\beta$

Dann ist $w := \alpha r \oplus [v[hn]]\beta$ mit $n = float$ eine valide Transition.

Löschen einer Stimme Sei $w = \alpha[hn]\beta$

Dann ist $w := \alpha\beta$ mit eine valide Transition.

Hinzufügen einer Delegation Analog der Stimme

Löschen einer Delegation Analog der Stimme

Delegationsprogrammierung

Context Delegation

A/B Tester

zu untersuchen

- wie werden neue anteile vergeben