TYPED FEATURE STRUCTURES AS DESCRIPTIONS

Paul John King*

Seminar für Sprachwissenschaft, Eberhard-Karls-Universität†

ABSTRACT

A description is an entity that can be interpreted as true or false of an object, and using feature structures as descriptions accrues several computational benefits. In this paper, I create an explicit interpretation of a typed feature structure used as a description, define the notion of a satisfiable feature structure, and create a simple and effective algorithm to decide if a feature structure is satisfiable.

1. INTRODUCTION

Describing objects is one of several purposes for which linguists use feature structures. A description is an entity that can be interpreted as true or false of an object. For example, the conventional interpretation of the description 'it is black' is true of a soot particle, but false of a snowflake. Therefore, any use of a feature structure to describe an object demands that the feature structure can be interpreted as true or false of the object. In this paper, I tailor the semantics of [King 1989] to suit the typed feature structures of [Carpenter 1992], and so create an explicit interpretation of a typed feature structure used as a description. I then use this interpretation to define the notion of a satisfiable feature structure.

Though no feature structure algebra provides descriptions as expressive as those provided by a feature logic, using feature structures to describe objects profits from a large stock of available computational techniques to represent, test and process feature structures. In this paper, I demonstrate the computational benefits of marrying a tractable syntax and an explicit semantics by creating a simple and effective algorithm to decide the satisfiability of a feature structure. Gerdemann and Götz's Troll type resolution system implements both the semantics and an efficient refinement of the satisfiability algorithm I present here (see [Götz 1993], [Gerdemann and King 1994] and [Gerdemann (fc)]).

2. A FEATURE STRUCTURE SEMANTICS

A signature provides the symbols from which to construct typed feature structures, and an interpretation gives those symbols meaning.

Definition 1. Σ *is a signature iff*
Σ is a sextuple $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$,
$\mathfrak{Q}$ is a set,
$\langle \mathfrak{T}, \preceq \rangle$ is a partial order,
$$\mathfrak{S} = \left\{ \sigma \in \mathfrak{T} \,\middle|\, \begin{array}{l} \text{for each } \tau \in \mathfrak{T}, \\ \text{if } \sigma \preceq \tau \text{ then } \sigma = \tau \end{array} \right\},$$
$\mathfrak{A}$ is a set,
$\mathfrak{F}$ is a partial function from the Cartesian product of $\mathfrak{T}$ and $\mathfrak{A}$ to $\mathfrak{T}$, and
for each $\tau \in \mathfrak{T}$, each $\tau' \in \mathfrak{T}$ and each $\alpha \in \mathfrak{A}$,
if $\mathfrak{F}(\tau, \alpha)$ is defined and $\tau \preceq \tau'$
then $\mathfrak{F}(\tau', \alpha)$ is defined, and
$\mathfrak{F}(\tau, \alpha) \preceq \mathfrak{F}(\tau', \alpha)$.

Henceforth, I tacitly work with a signature $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$. I call members of $\mathfrak{Q}$ states, members of $\mathfrak{T}$ types, $\preceq$ subsumption, members of $\mathfrak{S}$ species, members of $\mathfrak{A}$ attributes, and $\mathfrak{F}$ appropriateness.

Definition 2. *I is an interpretation iff*
I is a triple $\langle U, S, A \rangle$,
U is a set,
S is a total function from U to $\mathfrak{S}$
A is a total function from $\mathfrak{A}$ to the set of partial functions from U to U,
for each $\alpha \in \mathfrak{A}$ and each $u \in U$,
if $A(\alpha)(u)$ is defined
then $\mathfrak{F}(S(u), \alpha)$ is defined, and
$\mathfrak{F}(S(u), \alpha) \preceq S(A(\alpha)(u))$, and
for each $\alpha \in \mathfrak{A}$ and each $u \in U$,
if $\mathfrak{F}(S(u), \alpha)$ is defined
then $A(\alpha)(u)$ is defined.

Suppose that I is an interpretation $\langle U, S, A \rangle$. I call each member of U an object in I.

*The research presented in this paper was sponsored by Teilprojekt B4 "Constraints on Grammar for Efficient Generation" of the Sonderforschungsbereich 340 of the Deutsche Forschungsgemeinschaft. I also wish to thank Bob Carpenter, Dale Gerdemann, Thilo Götz and Jennifer King for their invaluable help with this paper.

†Wilhelmstr. 113, 72074 Tübingen, Germany. Email: king@sfs.nphil.uni-tuebingen.de.

Each type denotes a set of objects in I. The denotations of the species partition U, and S assigns each object in I the unique species whose denotation contains the object: object u is in the denotation of species σ iff $\sigma = S(u)$. Subsumption encodes a relationship between the denotations of species and types: object u is in the denotation of type τ iff $\tau \preceq S(u)$. So, if $\tau_1 \preceq \tau_2$ then the denotation of type τ_1 contains the denotation of type τ_2.

Each attribute denotes a partial function from the objects in I to the objects in I, and A assigns each attribute the partial function it denotes. Appropriateness encodes a relationship between the denotations of species and attributes: if $\mathfrak{F}\langle\sigma,\alpha\rangle$ is defined then the denotation of attribute α acts upon each object in the denotation of species σ to yield an object in the denotation of type $\mathfrak{F}\langle\sigma,\alpha\rangle$, but if $\mathfrak{F}\langle\sigma,\alpha\rangle$ is undefined then the denotation of attribute α acts upon no object in the denotation of species σ. So, if $\mathfrak{F}\langle\tau,\alpha\rangle$ is defined then the denotation of attribute α acts upon each object in the denotation of type τ to yield an object in the denotation of type $\mathfrak{F}\langle\tau,\alpha\rangle$.

I call a finite sequence of attributes a path, and write $\mathfrak{P}$ for the set of paths.

Definition 3. *P is the path interpretation function under I iff*
I is an interpretation $\langle U,S,A\rangle$,
P is a total function from $\mathfrak{P}$ to the set of partial functions from U to U, and
for each $\langle\alpha_1,\ldots,\alpha_n\rangle \in \mathfrak{P}$,
$P\langle\alpha_1,\ldots,\alpha_n\rangle$ is the functional composition of $A(\alpha_1),\ldots,A(\alpha_n)$.

I write P_I for the path interpretation function under I.

Definition 4. *F is a feature structure iff*
F is a quadruple $\langle Q,q,\delta,\theta\rangle$,
Q is a finite subset of $\mathfrak{Q}$,
$q \in Q$,
δ is a finite partial function from the Cartesian product of Q and $\mathfrak{A}$ to Q,
θ is a total function from Q to $\mathfrak{T}$, and
for each $q' \in Q$,
for some $\pi \in \mathfrak{P}$, π runs to q' in F,
where $\langle\alpha_1,\ldots,\alpha_n\rangle$ runs to q' in F iff
$\langle\alpha_1,\ldots,\alpha_n\rangle \in \mathfrak{P}$,
$q' \in Q$, and
for some $\{q_0,\ldots,q_n\} \subseteq Q$,
$q = q_0$,
for each $i < n$,
$\delta(q_i,\alpha_{i+1})$ is defined, and
$\delta(q_i,\alpha_{i+1}) = q_{i+1}$, and
$q_n = q'$.

Each feature structure is a connected Moore machine (see [Moore 1956]) with finitely many states, input alphabet $\mathfrak{A}$, and output alphabet $\mathfrak{T}$.

Definition 5. *F is true of u under I iff*
F is a feature structure $\langle Q,q,\delta,\theta\rangle$,
I is an interpretation $\langle U,S,A\rangle$,
u is an object in I, and
for each $\pi_1 \in \mathfrak{P}$, each $\pi_2 \in \mathfrak{P}$ and each $q' \in Q$,
if π_1 runs to q' in F, and
π_2 runs to q' in F
then $P_I(\pi_1)(u)$ is defined,
$P_I(\pi_2)(u)$ is defined,
$P_I(\pi_1)(u) = P_I(\pi_2)(u)$, and
$\theta(q') \preceq S(P_I(\pi_1)(u))$.

Definition 6. *F is a satisfiable feature structure iff*
F is a feature structure, and
for some interpretation I and some object u in I, F is true of u under I.

3. MORPHS

The abundance of interpretations seems to preclude an effective algorithm to decide if a feature structure is satisfiable. However, I insert *morphs* between feature structures and objects to yield an interpretation free characterisation of a satisfiable feature structure.

Definition 7. *M is a semi-morph iff*
M is a triple $\langle\Delta,\Gamma,\Lambda\rangle$,
Δ is a nonempty subset of $\mathfrak{P}$,
Γ is an equivalence relation over Δ,
for each $\alpha \in \mathfrak{A}$, each $\pi_1 \in \mathfrak{P}$ and each $\pi_2 \in \mathfrak{P}$,
if $\pi_1\alpha \in \Delta$ and $\langle\pi_1,\pi_2\rangle \in \Gamma$
then $\langle\pi_1\alpha,\pi_2\alpha\rangle \in \Gamma$,
Λ is a total function from Δ to $\mathfrak{S}$,
for each $\pi_1 \in \mathfrak{P}$ and each $\pi_2 \in \mathfrak{P}$,
if $\langle\pi_1,\pi_2\rangle \in \Gamma$ then $\Lambda(\pi_1) = \Lambda(\pi_2)$, and
for each $\alpha \in \mathfrak{A}$ and each $\pi \in \mathfrak{P}$,
if $\pi\alpha \in \Delta$
then $\pi \in \Delta$, $\mathfrak{F}(\Lambda(\pi),\alpha)$ is defined, and
$\mathfrak{F}(\Lambda(\pi),\alpha) \preceq \Lambda(\pi\alpha)$.

Definition 8. *M is a morph iff*
M is a semi-morph $\langle\Delta,\Gamma,\Lambda\rangle$, and
for each $\alpha \in \mathfrak{A}$ and each $\pi \in \mathfrak{P}$,
if $\pi \in \Delta$ and $\mathfrak{F}(\Lambda(\pi),\alpha)$ is defined
then $\pi\alpha \in \Delta$.

Each morph is the Moshier abstraction (see [Moshier 1988]) of a connected and totally well-typed (see [Carpenter 1992]) Moore machine with possibly infinitely many states, input alphabet $\mathfrak{A}$, and output alphabet $\mathfrak{S}$.

Definition 9. *M abstracts u under I iff*
M is a morph $\langle\Delta, \Gamma, \Lambda\rangle$,
I is an interpretation $\langle U, S, A\rangle$,
u is an object in I,
for each $\pi_1 \in \mathfrak{P}$ and each $\pi_2 \in \mathfrak{P}$,
$\langle\pi_1, \pi_2\rangle \in \Gamma$
iff $P_I(\pi_1)(u)$ is defined,
$P_I(\pi_2)(u)$ is defined, and
$P_I(\pi_1)(u) = P_I(\pi_2)(u)$, and
for each $\sigma \in \mathfrak{S}$ and each $\pi \in \mathfrak{P}$,
$\langle\pi, \sigma\rangle \in \Lambda$
iff $P_I(\pi)(u)$ is defined, and
$\sigma = S(P_I(\pi)(u))$.

Proposition 10. *For each interpretation I and each object u in I,*
some unique morph abstracts u under I.

I thus write of *the* abstraction of u under I.

Definition 11. *u is a standard object iff*
u is a quadruple $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle$,
$\langle\Delta, \Gamma, \Lambda\rangle$ is a morph, and
E is an equivalence class under Γ.

I write $\widetilde{U}$ for the set of standard objects, write $\widetilde{S}$ for the total function from $\widetilde{U}$ to $\mathfrak{S}$, where
for each $\sigma \in \mathfrak{S}$ and each $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle \in \widetilde{U}$,
$\widetilde{S}\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle = \sigma$
iff for some $\pi \in \mathrm{E}$, $\Lambda(\pi) = \sigma$,
and write $\widetilde{A}$ for the total function from $\mathfrak{A}$ to the set of partial functions from $\widetilde{U}$ to $\widetilde{U}$, where
for each $\alpha \in \mathfrak{A}$, each $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle \in \widetilde{U}$ and each $\langle\Delta', \Gamma', \Lambda', \mathrm{E}'\rangle \in \widetilde{U}$,
$\widetilde{A}(\alpha)\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle$ is defined, and
$\widetilde{A}(\alpha)\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle = \langle\Delta', \Gamma', \Lambda', \mathrm{E}'\rangle$
iff $\langle\Delta, \Gamma, \Lambda\rangle = \langle\Delta', \Gamma', \Lambda'\rangle$, and
for some $\pi \in \mathrm{E}$, $\pi\alpha \in \mathrm{E}'$.

Lemma 12. *$\langle\widetilde{U}, \widetilde{S}, \widetilde{A}\rangle$ is an interpretation.*

I write $\widetilde{I}$ for $\langle\widetilde{U}, \widetilde{S}, \widetilde{A}\rangle$.

Lemma 13. *For each $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle \in \widetilde{U}$, each $\langle\Delta', \Gamma', \Lambda', \mathrm{E}'\rangle \in \widetilde{U}$ and each $\pi \in \mathfrak{P}$,*
$P_{\widetilde{I}}(\pi)\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle$ is defined, and
$P_{\widetilde{I}}(\pi)\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle = \langle\Delta', \Gamma', \Lambda', \mathrm{E}'\rangle$
iff $\langle\Delta, \Gamma, \Lambda\rangle = \langle\Delta', \Gamma', \Lambda'\rangle$, and
for some $\pi' \in \mathrm{E}$, $\pi'\pi \in \mathrm{E}'$.

Proof. By induction on the length of π. ■

Lemma 14. *For each $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle \in \widetilde{U}$,*
if E is the equivalence class of the empty path under Γ
then the abstraction of $\langle\Delta, \Gamma, \Lambda, \mathrm{E}\rangle$ under $\widetilde{I}$ is $\langle\Delta, \Gamma, \Lambda\rangle$.

Proposition 15. *For each morph M,*
for some interpretation I and some object u in I,
M is the abstraction of u under I.

Definition 16. *F approximates M iff*
F is a feature structure $\langle Q, q, \delta, \theta\rangle$,
M is a morph $\langle\Delta, \Gamma, \Lambda\rangle$, and
for each $\pi_1 \in \mathfrak{P}$, each $\pi_2 \in \mathfrak{P}$ and each $q' \in Q$,
if π_1 runs to q' in F, and
π_2 runs to q' in F
then $\langle\pi_1, \pi_2\rangle \in \Gamma$, and
$\theta(q') \preceq \Lambda(\pi_1)$.

A feature structure approximates a morph iff the Moshier abstraction of the feature structure abstractly subsumes (see [Carpenter 1992]) the morph.

Proposition 17. *For each interpretation I, each object u in I and each feature structure F,*
F is true of u under I
iff F approximates the abstraction of u under I.

Theorem 18. *For each feature structure F,*
F is satisfiable iff F approximates some morph.

Proof. From propositions 15 and 17. ■

4. RESOLVED FEATURE STRUCTURES

Though theorem 18 gives an interpretation free characterisation of a satisfiable feature structure, the characterisation still seems to admit of no effective algorithm to decide if a feature structure is satisfiable. However, I use theorem 18 and *resolved feature structures* to yield a less general interpretation free characterisation of a satisfiable feature structure that admits of such an algorithm.

Definition 19. *R is a resolved feature structure iff*
R is a feature structure $\langle Q, q, \delta, \rho\rangle$,
ρ is a total function from Q to $\mathfrak{S}$, and
for each $\alpha \in \mathfrak{A}$ and each $q' \in Q$,
if $\delta(q', \alpha)$ is defined
then $\mathfrak{F}(\rho(q'), \alpha)$ is defined, and
$\mathfrak{F}(\rho(q'), \alpha) \preceq \rho(\delta(q', \alpha))$.

Each resolved feature structure is a well-typed (see [Carpenter 1992]) feature structure with output alphabet $\mathfrak{S}$.

Definition 20. *R is a resolvant of F iff*
R is a resolved feature structure $\langle Q, q, \delta, \rho\rangle$,
F is a feature structure $\langle Q, q, \delta, \theta\rangle$, and
for each $q' \in Q$, $\theta(q') \preceq \rho(q')$.

Proposition 21. *For each interpretation I, each object u in I and each feature structure F,*
F is true of u under I
iff some resolvant of F is true of u under I.

Definition 22. $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is rational iff for each* $\sigma \in \mathfrak{S}$ *and each* $\alpha \in \mathfrak{A}$,

if $\mathfrak{F}(\sigma, \alpha)$ *is defined*

then for some $\sigma' \in \mathfrak{S}$, $\mathfrak{F}(\sigma, \alpha) \preceq \sigma'$.

Proposition 23. *If* $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is rational then for each resolved feature structure* R, R *is satisfiable.*

Proof. Suppose that $R = \langle Q, q, \delta, \rho \rangle$ and β is a bijection from ordinal ζ to $\mathfrak{S}$. Let

$$\Delta_0 = \left\{ \pi \,\middle|\, \begin{array}{l} \text{for some } q' \in Q, \\ \pi \text{ runs to } q' \text{ in } R \end{array} \right\},$$

$$\Gamma_0 = \left\{ \langle \pi_1, \pi_2 \rangle \,\middle|\, \begin{array}{l} \text{for some } q' \in Q, \\ \pi_1 \text{ runs to } q' \text{ in } R, \text{ and} \\ \pi_2 \text{ runs to } q' \text{ in } R \end{array} \right\},$$

and

$$\Lambda_0 = \left\{ \langle \pi, \sigma \rangle \,\middle|\, \begin{array}{l} \text{for some } q' \in Q, \\ \pi \text{ runs to } q' \text{ in } R, \text{ and} \\ \sigma = \rho(q') \end{array} \right\}.$$

For each $n \in \mathbb{N}$, let

$$\Delta_{n+1} = \Delta_n \cup \left\{ \pi\alpha \,\middle|\, \begin{array}{l} \alpha \in \mathfrak{A}, \\ \pi \in \Delta_n, \text{ and} \\ \mathfrak{F}(\Lambda_n(\pi), \alpha) \text{ is defined} \end{array} \right\},$$

$$\Gamma_{n+1} = \Gamma_n \cup \left\{ \langle \pi_1\alpha, \pi_2\alpha \rangle \,\middle|\, \begin{array}{l} \alpha \in \mathfrak{A}, \\ \pi_1\alpha \in \Delta_{n+1}, \\ \pi_2\alpha \in \Delta_{n+1}, \text{ and} \\ \langle \pi_1, \pi_2 \rangle \in \Gamma_n, \end{array} \right\}, \text{ and}$$

$$\Lambda_{n+1} = \Lambda_n \cup \left\{ \langle \pi\alpha, \beta(\xi) \rangle \,\middle|\, \begin{array}{l} \alpha \in \mathfrak{A}, \\ \pi \in \Delta_n, \\ \pi\alpha \in \Delta_{n+1} \setminus \Delta_n, \text{ and} \\ \xi \text{ is the least ordinal} \\ \ \text{in } \zeta \text{ such that} \\ \ \mathfrak{F}(\Lambda_n(\pi), \alpha) \preceq \beta(\xi) \end{array} \right\}.$$

For each $n \in \mathbb{N}$, $\langle \Delta_n, \Gamma_n, \Lambda_n \rangle$ is a semi-morph. Let

$\Delta = \bigcup \{ \Delta_n \mid n \in \mathbb{N} \}$,

$\Gamma = \bigcup \{ \Gamma_n \mid n \in \mathbb{N} \}$, and

$\Lambda = \bigcup \{ \Lambda_n \mid n \in \mathbb{N} \}$.

$\langle \Delta, \Gamma, \Lambda \rangle$ is a morph that R approximates. By theorem 18, R is satisfiable. ■

Theorem 24. *If* $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is rational then for each feature structure* F,

F *is satisfiable iff* F *has a resolvant.*

Proof. From propositions 21 and 23. ■

5. A SATISFIABILITY ALGORITHM

In this section, I use theorem 24 to show how – given a rational signature that meets reasonable computational conditions – to construct an effective algorithm to decide if a feature structure is satisfiable.

Definition 25. $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is computable iff*

$\mathfrak{Q}$, $\mathfrak{T}$ *and* $\mathfrak{A}$ *are countable,*

$\mathfrak{S}$ *is finite,*

for some effective function `SUB`,

for each $\tau_1 \in \mathfrak{T}$ *and each* $\tau_2 \in \mathfrak{T}$,

if $\tau_1 \preceq \tau_2$

then $\texttt{SUB}(\tau_1, \tau_2) =$ *'true'*

otherwise $\texttt{SUB}(\tau_1, \tau_2) =$ *'false', and*

for some effective function `APP`,

for each $\tau \in \mathfrak{T}$ *and each* $\alpha \in \mathfrak{A}$,

if $\mathfrak{F}(\tau, \alpha)$ *is defined*

then $\texttt{APP}(\tau, \alpha) = \mathfrak{F}(\tau, \alpha)$

otherwise $\texttt{APP}(\tau, \alpha) =$*'undefined'.*

Proposition 26. *If* $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is computable then for some effective function* `RES`,

for each feature structure F,

$\texttt{RES}(F) =$ *a list of the resolvants of* F.

Proof. Since $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ is computable, for some effective function `GEN`,

for each finite $Q \subseteq \mathfrak{Q}$,

$\texttt{GEN}(Q) =$ a list of the total functions from Q to $\mathfrak{S}$,

for some effective function $\texttt{TEST}_1$,

for each finite set Q, each finite partial function δ from the Cartesian product of Q and $\mathfrak{A}$ to Q, and each total function θ from Q to $\mathfrak{T}$,

if for each $\langle q, \alpha \rangle$ in the domain of δ,

$\mathfrak{F}(\theta(q), \alpha)$ is defined, and

$\mathfrak{F}(\theta(q), \alpha) \preceq \theta(\delta(q, \alpha))$

then $\texttt{TEST}_1(\delta, \theta) =$ 'true'

otherwise $\texttt{TEST}_1(\delta, \theta) =$ 'false',

and for some effective function $\texttt{TEST}_2$,

for each finite set Q, each total function θ_1 from Q to $\mathfrak{T}$ and each total function θ_2 from Q to $\mathfrak{T}$,

if for each $q \in Q$, $\theta_1(q) \preceq \theta_2(q)$

then $\texttt{TEST}_2(\theta_1, \theta_2) =$ 'true'

otherwise $\texttt{TEST}_2(\theta_1, \theta_2) =$ 'false'.

Construct `RES` as follows:

for each feature structure $\langle Q, q, \delta, \theta \rangle$,

set $\Sigma_{\text{in}} = \texttt{GEN}(Q)$ and $\Sigma_{\text{out}} = \langle \rangle$

while $\Sigma_{\text{in}} = \langle \rho, \rho_1, \ldots, \rho_i \rangle$ is not empty

do set $\Sigma_{\text{in}} = \langle \rho_1, \ldots, \rho_i \rangle$

if $\texttt{TEST}_1(\delta, \rho) =$ 'true',

$\texttt{TEST}_2(\theta, \rho) =$ 'true', and

$\Sigma_{\text{out}} = \langle \rho'_1, \ldots, \rho'_j \rangle$

then set $\Sigma_{\text{out}} = \langle \rho, \rho'_1, \ldots, \rho'_j \rangle$

if $\Sigma_{\text{out}} = \langle \rho_1, \ldots, \rho_n \rangle$

then output $\langle \langle Q, q, \delta, \rho_1 \rangle, \ldots, \langle Q, q, \delta, \rho_n \rangle \rangle$.

`RES` is an effective algorithm, and

for each feature structure F,

$\texttt{RES}(F) =$ a list of the resolvants of F.

■

Theorem 27. *If* $\langle \mathfrak{Q}, \mathfrak{T}, \preceq, \mathfrak{S}, \mathfrak{A}, \mathfrak{F} \rangle$ *is rational and computable then for some effective function* `SAT`*,*

for each feature structure F*,*

if F *is satisfiable*

then $\mathtt{SAT}(F) =$ *'true*

otherwise $\mathtt{SAT}(F) =$ *'false'.*

Proof. From theorem 24 and proposition 26. ■

Gerdemann and Götz's Troll system (see [GÖTZ 1993], [GERDEMANN AND KING 1994] and [GERDEMANN (FC)]) employs an efficient refinement of `RES` to test the satisfiability of feature structures. In fact, Troll represents each feature structure as a disjunction of the resolvants of the feature structure. Loosely speaking, the resolvants of a feature structure have the same underlying finite state automaton as the feature structure, and differ only in their output function. Troll exploits this property to represent each feature structure as a finite state automaton and a set of output functions. The Troll unifier is closed on these representations. Thus, though `RES` is computationally expensive, Troll uses `RES` only during compilation, never during run time.

References

[CARPENTER 1992] Robert Carpenter *The logic of typed feature structures.* Cambridge tracts in theoretical computer science 32. Cambridge University Press, Cambridge, England. 1992.

[GERDEMANN (FC)] Dale Gerdemann. *Troll: type resolution system, user's guide.* Sonderforschungsbereich 340 technical report. Eberhard-Karls-Universität, Tübingen, Germany. Forthcoming.

[GERDEMANN AND KING (1994)] Dale Gerdemann and Paul John King. *The correct and efficient implementation of appropriateness specifications for typed feature structures.* In these proceedings.

[GÖTZ 1993] Thilo Götz. *A normal form for typed feature structures.* Master's thesis. Eberhard-Karls-Universität, Tübingen, Germany. 1993.

[KING 1989] Paul John King. *A logical formalism for head-driven phrase structure grammar.* Doctoral thesis. The University of Manchester, Manchester, England. 1989.

[MOORE 1956] E. F. Moore. 'Gedanken experiments on sequential machines'. In *Automata Studies.* Princeton University Press, Princeton, New Jersey, USA. 1956.

[MOSHIER 1988] Michael Andrew Moshier. *Extensions to unification grammar for the description of programming languages.* Doctoral thesis. The University of Michigan, Ann Arbor, Michigan, USA. 1988.