UMA ANÁLISE EVOLUTIVA E TÉCNICA DO SISTEMA ESPECIALISTA HEARSAY

Fernando Beletti, Filipe Marques Barbosa, Lucas Wesley de Lima, Thomás Carvalho Sales Pitombeira, Vitor Augusto Santos Silva, Shigueo Nomura*
Universidade Federal de Uberlândia, Faculdade de Engenharia Elétrica, Uberlândia – MG
*Universidade Federal de Uberlândia, Faculdade de Computação, Uberlândia - MG
fbeletti@gmail.com, flpmarques13@gmail.com, lucaswesley10@gmail.com, thomas.com@hotmail.com, vitoraugusto.ufu@gmail.com, shigueonomura@gmail.com

Resumo **- Este artigo apresenta uma análise evolutiva e uma exemplificação do Hearsay, que é um sistema especialista para reconhecimento de voz baseado em sistema de quadro-negro. O Hearsay utiliza diversos recursos da fonte de conhecimento que está implementada em um banco de dados único. Neste trabalho, são analisadas a sua arquitetura, representação do conhecimento e as regras para seu próprio mecanismo de inferência. Além disso, um exemplo de aplicação do sistema é apresentado. Conclui-se que a evolução do sistema inteligente através de diversas versões trouxe uma melhoria significativa nos resultados do mecanismo de inferência para o reconhecimento de voz.**

Palavras-Chave **– Hearsay, mecanismo de inferência, reconhecimento de voz, sistema de quadro-negro, sistema especialista.**

AN EVOLUTIVE AND TECHNICAL ANALYSIS FOR HEARSAY EXPERT SYSTEM

Abstract **- This paper presents an evolutive analysis and an exemplification of Hearsay which is an expert system for speech recognition based on blackboard system. Hearsay utilizes various resources from the knowledge source that is implemented in a unique database. In this work, we have analyzed its architecture, knowledge representation and rules for its own inference mechanism. Moreover, an application example of the system is presented. We have concluded that the intelligent system evolution through several versions have led to a significant improvement on the results of inference mechanism for the speech recognition.**

Keywords **– Blackboard system, expert system, Hearsay, inference mechanism, speech recognition.**

XIII CEEL - ISSN 2178-8308
12 a 16 de Outubro de 2015
Universidade Federal de Uberlândia - UFU
Uberlândia - Minas Gerais - Brasil

I. INTRODUÇÃO

O Hearsay é um sistema especialista (SE) elaborado com o intuito de permitir a identificação de palavras com base na interpretação de sinais. Para tal, ele se baseia em diversas estruturas de comparação de palavras e de sons convertidos para sinais elétricos, a fim de decidir qual a melhor opção a ser apresentada para o usuário objetivando ter o menor erro possível.

Para que um sistema seja capaz de desempenhar a função de reconhecer a fala, alguns conceitos relacionados a ela devem ser considerados, tais como fonema, semântica, escolha das palavras, formulação das frases, etc. Para exemplificar melhor a situação pela qual o sistema passa, podemos imaginar uma conversa entre duas pessoas. Para que a conversa possa fluir perfeitamente é necessário que os interlocutores entendam o que está sendo dito por ambas as partes. Essa questão da compreensão é justamente a função do Hearsay e também do cérebro humano, que capta as ondas sonoras recebidas pelo ouvido e as interpretam de acordo com as informações adquiridas ao longo do tempo, permitindo que um ser humano seja capaz de reconhecer os fonemas, sílabas, palavras, pausas na fala e frases que foram ditos por outra pessoa. O que torna este processo tão complicado de ser implementado em um sistema de computador é a grande variedade de palavras em um idioma e as diversas formas de acentuação, que alteram a forma de pronúncia de algumas letras. Sendo assim, se o sistema não for bem elaborado, é possível que uma palavra que não se encaixe em uma sentença seja escolhida no lugar de outra apenas pelo fato de ambas possuírem pronúncias semelhantes.

II. CLASSES DE SISTEMAS ESPECIALISTAS

Os sistemas especialistas (SEs) em geral são classificados de acordo com seus domínios de conhecimento, mecanismos de inferência ou mecanismos de classificação de conhecimento. Mas, uma opção é a classificação de acordo com sua usabilidade, sendo assim separados em três classes de sistemas especialistas [1]:

- Sistemas de classe 1: São sistemas que possuem a maior aplicação comercial e maior aceitação pelos usuários. Isso porque são sistemas que possuem o mínimo de interação com o usuário e possuem um tratamento direto e não controverso do conhecimento,

alcançando um resultado concreto. Geralmente possuem domínios de conhecimento bem definidos e focados [1];

- Sistemas de Classe 2: Os sistemas dessa classe são SE que possuem boa performance comparável aos de classe 1, mas que não possuem boa aceitação do usuário. Isso porque não expõem claramente seus cálculos ao usuário, muitas vezes ignorando algumas predefinições e apresentando suas próprias conclusões. Talvez, o principal motivo para sua pouca aceitação é sua interface com o usuário, e sua grande aplicação em situações onde o resultado é controverso, não sendo errado, mas podendo gerar uma consequência não desejada como, por exemplo, em aplicações médicas [1];
- Sistemas de Classe 3: Os SEs de classe 3 são sistemas que não possuem boa aceitação de seu "público alvo". Isso porque não conseguem alcançar resultados completamente corretos e assim não possuem boa aplicação comercial. Geralmente isso ocorre por seus objetivos serem ambiciosos e inovadores, sendo assim bons contribuintes no avanço de novas técnicas. Também precisam utilizar várias e amplas fontes de conhecimento para seus cálculos [1].

O sistema Hearsay é caracterizado como um sistema de classe 3 [1].

III. ARQUITETURA DO HEARSAY-III

A. O Banco de Dados Relacional Básico

O Hearsay-III é construído sobre uma base composta por um sistema de banco de dados e suas correspondentes instalações de controle. A linguagem de banco de dados é chamada AP3 e é incorporado em Interlisp (ambiente de programação construído com base na linguagem Lisp e é uma ferramenta de correção automática de erros simples). Como será visto nas secções subsequentes, o Hearsay-III depende criticamente das ferramentas oferecidas pelo AP3 [2].

A base de dados da AP3 é semelhante em estrutura aos disponíveis na Linguagem PLANNER-like. Também oferece grande facilidade de digitação em declaração, recuperação e passagem de parâmetros na chamada de funções. A facilidade de digitação no AP3 está disponível para um usuário Hearsay-III na modelagem de domínio da aplicação para ser utilizado vantajosamente dentro do sistema em si. O quadro-negro do Hearsay-III e todos os dados e estruturas do sistema acessíveis ao público, são representados na base de dados da AP3. Anotações adicionais exigidas pelas fontes de conhecimento da aplicação podem também ser colocadas no banco de dados da AP3, juntamente com os gatilhos necessários para sua ativação [2].

O sistema de banco de dados da AP3 também oferece ferramentas para inferência, regras e restrições. Estas ferramentas, além de serem utilizadas na aplicação do próprio Hearsay-III, também estão disponíveis para o usuário codificar as relações globais de domínios dependentes [2].

B. Estrutura do Quadro-Negro

A estrutura de quadro-negro do Hearsay é apresentada na Figura 1.

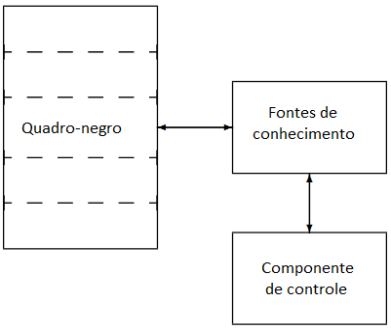

Fig. 1. Componentes da estrutura do quadro-negro

O quadro-negro é um banco de dados global contendo todo o conteúdo de entrada, soluções parciais e outras informações de vários estados de resolução de problemas [3].

Ele possui os seguintes recursos:

- Memória compartilhada de dados de entrada ainda não trabalhadas, soluções parciais, alternativas e finais [3];
- Um meio de comunicação e armazenamento [3];
- Um mecanismo de disparo para as fontes de conhecimento [3].

As aplicações do quadro-negro tendem a ter elaboradas estruturas, com múltiplos níveis de análise ou abstração [3].

Ocasionalmente, o sistema contém subsistemas que se comunicam usando um banco de dados global que é apresentado incorretamente como sistema de quadro-negro (um conjunto de rotinas da linguagem FORTRAN usando COMMON é um exemplo extremo usando este raciocínio). O verdadeiro sistema de quadro-negro envolve uma interação fechada entre as fontes de conhecimento e um mecanismo separado de controle [3].

1) KSs (knowledge sources – fontes de conhecimento)

Cada KS é separada e independente uma da outra. Ela não precisa saber da especialidade ou mesmo da existência de qualquer outra fonte. Entretanto, ela precisa ser capaz de entender o estado do processo da resolução do problema e uma representação de informações relevantes no quadro-negro [3].

Todas as KSs sabem das condições sobre as quais podem contribuir para a solução e na hora apropriada tentar fornecer informação para ajudar a direcionar a solução do problema. Este conhecimento que cada KS tem sobre quando contribuir para o processo da resolução do problema é conhecido como condição de gatilho [3].

KSs são muito mais segmentadas do que papéis individuais de sistemas especialistas. Enquanto estes sistemas trabalham disparando uma regra em resposta a um estímulo, o sistema de quadro-negro trabalha disparando um módulo de conhecimento inteiro ou KS [3].

As fontes de conhecimento não são agentes ativos no sistema do quadro-negro. Pelo contrário, a ativação das fontes (às vezes chamadas de instâncias de conhecimento) são entidades ativas competindo pela execução de recursos. A ativação da KS é a combinação do seu conhecimento e um

contexto de um gatilho específico. A distinção entre KSs e a sua ativação é importante em aplicações onde numerosos eventos fazem o gatilho da mesma fonte de conhecimento. Nestes casos, decisões de controle fazem a escolha entre aplicações particulares de KSs com mesmo conhecimento (focando nos dados apropriados ao contexto), mais do que entre diferentes KSs (focando no conhecimento apropriado para aplicar). Fontes de conhecimento são repositórios de conhecimento e a ativação da KS implica em ativação do processo [3].

2) Componente de controle

Um mecanismo de controle explícito direciona o processo de resolução do problema permitindo KSs a responder oportunamente a mudanças no banco de dados do quadro-negro. Na base do estado do quadro-negro e na marcação do gatilho das KSs, o mecanismo de controle escolhe o curso da ação a ser tomada [3].

O sistema de quadro-negro usa um estilo de raciocínio incremental: a solução do problema é construída um passo de cada vez. E a cada passo o sistema pode [3]:

- Executar qualquer KS engatilhada [3];
- Escolher um foco diferente de atenção, com base no estado da solução [3].

Sobre uma aproximação típica de controle, a corrente de ativação de KS sendo executada gera eventos que são contribuições feitas no quadro-negro. Estes eventos são mantidos (e possivelmente ranqueados) até que a execução da ativação da KS é completada. Neste ponto, o componente de controle usa eventos para disparar e ativar as KSs. A ativação da fonte de conhecimento é ranqueada e a ativação mais apropriada é selecionada para execução. Este ciclo continua até que o problema seja resolvido [3].

IV. REPRESENTAÇÃO DO CONHECIMENTO

A. Limitações do Modelo Hearsay-I

No Hearsay-I, a representação do conhecimento é explorada apenas no nível da palavra. Assim, quando se aplica o paradigma de hipótese e testes, a representação do conhecimento é uniforme, mas apenas diz respeito à cooperação a este nível. Isto se torna um problema, pois torna o modelo limitado, uma vez que não podemos relacionar cada elemento com os dados em análise de uma forma global, isto é, não há exatamente uma análise fora do nível da palavra, cada forma é analisada por si só e as KSs não conseguem cooperação plena, tornando uma aproximação além do nível da palavra algo extremamente complicado e ineficiente [4].

B. Evolução do Modelo e Representação Global

Com base nos problemas observados, viu-se a necessidade de evolução do modelo para uma estrutura mais geral, uniforme e natural para representar e tornar a operação de reconhecimento das expressões mais dinâmica e efetiva [4].

Na segunda versão do SE, o conhecimento passa a ser implementado em uma estrutura singular e uniforme, porém baseada em três dimensões. Isto é, uma dimensão representa a informação em diversos níveis, além do nível básico da palavra, incluindo análise léxica, fonética e semântica; a segunda dimensão representa o tempo da fala e a terceira dimensão contém a hipótese alternativa para um nível particular em um determinado tempo [4].

Por exemplo, na frase “Fui à escola de dia”, no tempo relativo ao vocábulo “dia”, uma hipótese alternativa para o nível da palavra seria “noite” [4].

Conceitualmente, existe uma maneira simples e uniforme de dinamicamente relacionar as hipóteses de um nível de conhecimento, com suas hipóteses alternativas de mesmo nível e ainda às hipóteses de outros níveis de conhecimento da mesma estrutura. Essas análises e relacionamentos resultam em um grafo de E/OU com suas devidas modificações, que são responsáveis por prover as relações de cooperação que são temporárias, resultando nas relações de dependência seletiva [4].

Assim, verifica-se que o sistema evoluiu, buscando expandir o esquema de hipótese e teste para todos os níveis de reconhecimento das expressões, causando relações de dependência e cooperação, fugindo do antigo esquema em que cada palavra era avaliada em seu nível de maneira isolada. [4]

Cada expressão tem características que variam dependendo de quem fala, de sua idade, sexo e condições físicas. Além disso, existem as características do ambiente, como ruídos do ambiente e características do transdutor, como resposta de frequência do microfone. Portanto, o sistema tenta se adaptar para encontrar a solução correta, dadas as inúmeras variações que as expressões podem apresentar, utilizando uma tabela que contém os parâmetros padrões (formas de onda) para vários sons expressos em um contexto neutro (sem ruídos) [4].

O conhecimento é utilizado em diversos momentos, relacionando a estrutura silábica em segmentações. Para cada segmento, o conhecimento é utilizado para filtrar a voz, ruídos e junções silábicas rotulando cada segmento. Desta forma, os sons nas expressões são baseados em certos pontos, como vogais fortes, silabares, pausas e entonação. Estes pontos são tomados para a análise léxica, tornando a tradução das expressões algo recursivo. Isto é, dada uma palavra hipotética, baseada na análise fonética, essa palavra é avaliada léxica e semanticamente no conjunto da expressão. Caso não apresente sentido real e confiável globalmente, são utilizadas novas KSs para a solução do problema, com procedimentos mais sofisticados, buscando traduzir a expressão da melhor forma. Isto ocorre, filtrando possíveis situações relacionadas a verbos, adjetivos e substantivos, devido à suas posições na expressão. Entretanto, é possível que se chegue a uma ambiguidade que deve ser resolvida da melhor forma ao se relacionar todos os níveis [4].

Com a recursividade é possível que sejam alteradas palavras que já haviam sido definidas em algum momento do reconhecimento para que o melhor resultado seja alcançado. As técnicas de inferência para o reconhecimento não são lineares. Sendo assim, é possível através da interpretação léxico-semântica encontrar possíveis pontuações. Além disso, devido ao contexto de utilização do SE, a cada segmento reconhecido da expressão podem ser excluídas ou

adicionadas possíveis KSs a serem utilizadas no processo total de inferência, conforme descrito a seguir [5].

V. TÉCNICAS DE INFERÊNCIA

Para que a inferência ocorra com o menor número possível de erros, o Hearsay necessita de várias bases de conhecimento, que nada mais são do que conjuntos de informações inseridas no sistema por um especialista, como por exemplo, um fonoaudiólogo e/ou um professor de gramática [6].

O Hearsay utiliza uma técnica de inferência conhecida como Análise por Síntese. Esta técnica consiste em subdividir o objetivo principal em vários objetivos secundários e alcançar cada um destes, um após o outro, até o momento em que o objetivo final é alcançado. Sendo assim, podemos dividir o objetivo principal do Hearsay, que é reconhecer frases faladas pelo usuário, em: frases formadas, sequência de palavras, grupos de palavras, sílabas, segmentos (letras) e parâmetros (fonemas), conforme se apresentam na Figura 2. Assim, como existem sete objetivos, contando o objetivo principal e os secundários, há sete níveis contendo uma ou mais bases de conhecimento que serão ativados em sequência. É possível que uma KS seja utilizada em mais de um nível [6].

Quando o usuário interagir com o sistema, isto é, falar alguma coisa, o sinal sonoro emitido será inserido no sistema e passará por um processo de digitalização, processo este que tem o intuito de facilitar a análise do sinal por parte do Hearsay. O sinal, depois de ser digitalizado, é dividido em várias partes e cada parte é comparada com todos os sinais que representam os fonemas de um determinado idioma contido na base de conhecimento, permitindo que o sistema escolha todos os fonemas que ele julga estar presente no sinal enviado pelo usuário, e assim, levando essas informações para o próximo nível [6].

Com uma determinada quantidade de fonemas selecionados, o sistema passa a utilizar outra base de conhecimento, na qual estão presentes as letras, nas suas varias formas de pronúncia. Mais uma vez, o Hearsay seleciona algumas letras que podem estar contidas na mensagem original, e então o próximo nível é liberado [6].

Tendo as possíveis letras, o software pode agora tentar construir sílabas. Todas as combinações possíveis são montadas de acordo com as informações provenientes dos dois níveis anteriores. Esse novo grupo de sílabas será analisado no próximo nível, no qual é usada a base de conhecimento onde estão presentes as mais diversas palavras [6].

Posteriormente o sistema começa a agrupar as sílabas em todas as possíveis palavras que podem ter sido ditas pelo usuário. Escolhidas as palavras, o sistema é capaz de produzir sequências de palavras utilizando a KS de um nível acima, mesmo que essas sequências de palavras não façam sentido [6].

Após o nível em que as sequências de palavras são criadas o nível de formulação de frases é ativado. Neste nível, as bases de conhecimento envolvidas analisam se há sentido nas sequências de palavras criadas pelo nível anterior. Se as sequências não tenham nenhum nexo elas são descartadas. Caso contrário, são enviadas para análise no próximo e último nível [6].

Finalmente, no último nível, após a coleta e seleção de todos os dados citados anteriormente, o sistema avalia todas as hipóteses criadas e escolhe a melhor delas [6].

Os sete níveis citados são representados na Figura 2 [6]:

1) Aquisição do sinal, extração e segmentação de parâmetros:

- SEG: Digitaliza o sinal, mede os parâmetros, e produz as classes de fonemas.

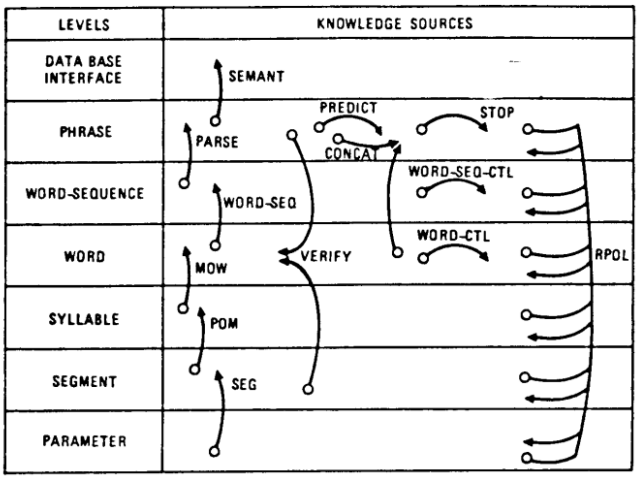

Fig. 2. Níveis de bases de conhecimento [6]

2) Área das palavras:

- POM: Cria classes de sílabas a partir dos fonemas.
- MOW: Cria possíveis palavras a partir das sílabas.
- WORD-CTL: Controla o número de palavras criadas.

3) Geração de frases:

- WORD-SEQ: Cria sequências de palavras com potencial para serem frases finais a partir das palavras criadas.
- WORD-SEQ-CTL: Controla o número de sequências de palavras criadas.
- PARSE: Analisa as sequências de palavras e cria frases com potencial.

4) Extensão das frases:

- PREDICT: Prevê todas as palavras possíveis que possam sintaticamente preceder ou seguir uma determinada frase.
- VERIFY: Avalia a consistência entre as hipóteses de segmento e um par de palavras-frase.
- CONCAT: Cria uma frase hipótese a partir de um par de palavras-frase, verificada a contiguidade.

5) Classificação e interpretação:

- RPOL: Avalia a credibilidade de cada nova hipótese ou hipótese modificada.

- STOP: Determina o momento de parada do processo.
- SEMANT: Fonte de conhecimento responsável por buscar a resposta final do problema [2], [6] e [7].

VI. EXEMPLO DE APLICAÇÃO

Para ilustrar o uso do Hearsay-III como uma linguagem de implementação para sistemas especialistas descreve-se aqui a implementação do Jitterer, um sistema de resolução de problemas. O problema endereçado pelo Jitterer é uma transformação automática da árvore de análise do programa. Os mapas do Jitterer transformam a árvore de análise (estado inicial) em uma nova árvore de análise (o estado objetivado) pela aplicação de uma sequência de transformações de preservação equivalente. A árvore de análise inicial, o intermediário e o final, gerado pela transformação sequencial é chamada de *programa de desenvolvimento de estados.* A descrição do estado objetivado é fornecida pelo usuário [2].

O Jitterer é um componente de um sistema *Implementação Transformacional* o qual permite ao usuário semi automaticamente refinar e otimizar uma especificação de alto nível do programa em uma implementação eficiente. Um exemplo de etapa de otimização *Implementação Transformacional* que o usuário pode tentar é a mesclagem de dois loops enumerados conjuntamente. A aplicação de uma transformação gera um novo, semanticamente equivalente, estado de desenvolvimento do programa [2].

A. Representação do Espaço/Estado

O projeto do Jitterer requer dois espaços adicionais:

- O espaço de desenvolvimento do programa, gerado pelas aplicações de transformação e representando vários estados de desenvolvimento,
- O espaço de raciocínio, gerado pelo primeiro melhor encadeamento para trás procurando e representando caminhos de soluções parciais e objetivos/sub- objetivos relacionados.

Usando-se a classe única dos mecanismos do Hearsay–III, a classe de Unidades-Dominantes pode ser subdividida em *Reasoning-Units* e *Developed Units*, e assim implementa-se dois espaços segmentando o domínio do quadro-negro [2].

B. Representação de Transformação

Cada transformação é implementada como um domínio KS (doravante, transformação KS). Pois no controle de encadeamento para trás do Jitterer, o gatilho da transformação KS corresponde à porção da ação (traduzido para coincidir com objetivos do espaço de raciocínio) da transformação correspondente. O código imediato da transformação KS é responsável pela criação, como submetas, do padrão LHS (left-hand-side pattern ou padrão acessor esquerdo) a ser correspondido e permitindo condições para ser estabelecido. Quando executado cria um novo contexto e faz as modificações apropriadas [2].

C. Conhecimento do Controle

Como descrito anteriormente, o Jitterer usa regras baseadas na seleção de conhecimento para controlar a busca. Cada regra de seleção é implementada como uma programação KS. Por exemplo, uma regra de seleção, trata o desejo de transformação como um efeito adicional que produz os efeitos da Figura 3 que mostra a forma da programação KS por uma instância de uma regra sabendo que a transformação que desdobra a função na linha foi um efeito adicional prejudicial enfraquecendo a estrutura do programa [2].

```
(Declare-SKS structural-flattening (Op)
   Trigger: (AND (Competi ngOperator OP)
                 (SideEffect Op UnfoldsFunction))
   immediate Code: Operator-orderinglevel
   Body: (Dec reaseDes i rabilityOP))
```

Fig. 3. Programação KS por uma instância de uma regra [2]

A Figura 4 mostra a programação KS dessa regra que verifica a existência de caminhos de desenvolvimento *PathThreshold*. Note que na avaliação de um código imediato de dois espaços de programações KSs, seus registros de ativação correspondem nos níveis de programação distintos. A base de programação do Jitterer fornece a *Path-state-change-level*, prioridade sobre o *Operator-ordering-level*. Por isso a suspenção do caminho é tratada antes de pedir a transformação [2].

```
(Declare-SKS lengthy-oath (Path)
  Trigger:
    (AND
       (CompetingPath Path)
       (> (CurrentPathLength Path) PathThreshold))
  immediate Code: Path-state-change-level
  Body: (MerkAsSuspended Path))
```

Fig. 4. Programação da regra que verifica a existência de caminhos de desenvolvimento [2]

As regras de seleção de referência determinam os recursos da solução de problemas. Esta informação é armazenada como uma ligação de estruturas de raciocínio auxiliares para uma unidade relevante na programação e no domínio de quadro-negro [2].

D. Uso de Rotinas Aceitáveis

A programação KS do Jitterer normalmente denota a dificuldade de arquivar uma meta particular, anexando informações apropriadas para a estrutura auxiliar de metas [2].

O Jitterer aplica-se a um conjunto de normalização e simplificação de regras cada vez que um novo estado de desenvolvimento do programa é gerado. Esse processo de limpeza tem sido implementado no Hearsay-III através da aceitação *Canonicalizer* (processo para converter dados que tem mais de uma representação possível na forma "normal" ou canônica). Cada tipo de nó (classe única) de uma árvore de análise tem uma *Canonicalizer* associada: cada *Canonicalizer* incorpora o conjunto de regras de limpeza por este tipo de nó. Ele é responsável pela transformação, mudando a árvore de análise para marcar os nós relevantes (unidades) para reaceitação [2].

VII. COMPARATIVO ENTRE AS VERSÕES DO HEARSAY

A. HSI – Hearsay Versão 1

Esta foi a primeira versão desse sistema. Sua principal característica é que o tratamento da voz seja feito de certa forma entre palavras, ou seja, cada palavra é tratada de forma completa sem dados externos ou recursividade no tratamento ou escolha de qual hipótese é a mais viável [4].

Seu sistema de inferência aciona cada KS em uma sequência de chamada, hipótese e teste. A rotina de chamada seleciona as KSs que possuem conhecimentos que contribuem na formulação da hipótese que será tratada. A rotina de hipótese define qual KS é mais confiável para então contribuir na hipótese e assim gera uma lista de possíveis palavras que se adequam à palavra recebida. A rotina de teste verifica qual das palavras geradas pela KS melhor se adequam gerando uma sentença parcial. A base de dados global seleciona a sentença mais promissora de cada KS reiniciando o ciclo de chamada, hipótese e teste [4].

B. HSII – Hearsay Versão 2

Na sua segunda versão, o sistema ampliou seu mecanismo de tratamento e na dinâmica de representação [4].

Agora as informações da fala são recebidas de forma tridimensional onde os diferentes tipos de conhecimentos são tratados de maneira uniforme recebendo os níveis de informação (ex.: léxica, fonética,...), tempo de fala e uma hipótese alternativa em um nível e tempo particular [4].

O principal objetivo dessa segunda versão foi ampliar sua esfera de observação para qualquer área do conhecimento e generalizando a rotina de chamada, hipótese e teste. A principal mudança foi feita na rotina de hipótese onde esta pode ser feita de forma horizontal ou vertical. Na forma horizontal, o sistema procura KSs relacionadas ao contexto da palavra em questão em mesmo nível. Quando a hipótese é feita de forma vertical, o nível é alterado, procurando-se palavras relacionadas para hipóteses em outros níveis do conhecimento [4].

E esse novo modelo de tratamento também oferece uma opção de questionamento da hipótese, que foi definida como a mais adequada. Assim gerando uma recursividade que trata o resultado semântico da hipótese, verificando se o resultado obtido tem sentido verdadeiro como frase e assim adaptando a hipótese e seus resultados [4].

C. HSIII – Hearsay Versão 3

A terceira versão do Hearsay foi o objeto de estudo principal deste trabalho. Em suma, a terceira versão trouxe como principais avanços frente às suas versões anteriores, a ampliação das informações de fala recebidas e a velocidade de tratamento de informações. Consequentemente, possibilitou uma ainda maior eficiência para o sistema.

VIII. CONCLUSÃO

Da análise realizada neste trabalho, conclui-se que, com a evolução do sistema Hearsay envolvendo a mudança nos métodos aplicados para o mecanismo de inferência ao longo das três versões deste sistema que se sucederam, resultados bastante satisfatórios passaram a ser alcançados para o reconhecimento eficiente de voz.

Ainda que o Hearsay não seja um sistema com aplicação exclusiva, sendo utilizado como solução de variados problemas, o sistema é confiável, apresentando um nível de acerto de interpretações da ordem de 90% conforme os experimentos realizados.

AGRADECIMENTOS

Agradecemos ao Prof. Dr. Shigueo Nomura que nos ministrou a disciplina de Inteligência Artificial na Universidade Federal de Uberlândia, pelo incentivo e apoio na realização deste trabalho.

REFERÊNCIAS

[1] A. Kandel, *Fuzzy Expert Systems*, CRC Press, 2000.

[2] L. D. Erman, P. E. London, S. F. Fickas, "The design and an example use of Hearsay-MI", *IJCAI-81 Vancouver Canada*, vol. 1, pp. 409-415, August 1981.

[3] D. D. Corkill, "Blackboard Systems", *AI Expert*, vol. 6, no. 9, January 1991.

[4] V. R. Lesser, R. D. Fennel, L. D Erman, D. R. Reddy "Organization of the Hearsay II Speech Understanding System", *IEEE transactions on acoustics, speech, and signal processing*, vol. ASSP-23, no. 1, February 1975.

[5] D. R. Reddy, L.D Erman, R.O Fennel, R. B. Neely "The Hearsay speech understanding system: An Example of the Recognition Process", *IEEE transactions on computers*, vol. C-25, no. 4, April 1976.

[6] L. D. Erman, F. Hayes-Roth, V. R. Lesser, D. R. Reddy, "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty", *Journal ACM Computing Surveys (CSUR)*, vol. 12, no. 2, pp. 213-253, June 1980.

[7] D. McCracken, *A Production System Version of the Hearsay-II Speech Understanding System*, Carnegie-Mellon University Computer Science Speech Group, April 1978.