به‌نام‌خدا

بررسی ابزار ایجاد سیستم‌های خبره فارسی
مبتنی بر شبکه معنائی .

اردوان مجیدی
دانشگاه شهید بهشتی
مرکز محاسبات- مرکزتحقیقات وساخت‌نرم‌افزار
دکتر محسن صدیقی مشکنانی
دانشگاه شهید بهشتی
دانشکده مهندسی برق و کامپیوتر

تذکر : این متن به دلیل تبدیل از محیط یک ویراستار دیگر، دارای اشکالات و نواقص صفحه بندی است و مورد ویرایش مجدد قرار نگرفته است .

چکیده

قسمت اعظم زمان طراحی و ایجاد یک سیستم خبره صرف طراحی و پیاده سازی روالهای عمل کننده در ایجاد و استفاده از پایگاه دانش و روالهای جستجو و توضیح می گردد و این موضوع علاوه بر افزایش زمان ایجاد، ذهن طراح را از تمرکز برموضوع اصلی خبرگی باز می دارد و به سطوح زیرین برنامه متوجه می کند.

این مقاله ابزاری را مورد مطالعه قرار می دهد که امکان ایجاد پایگاه دانش پویا مبتنی بر شبکه معنائی و مکانیزم های جستجو در پایگاه و نیز سیستم توضیح با امکانات محدود فارسی را در اختیار برنامه‌نویسان قرار می دهد.

این ابزار در عین حال قابلیت پیمایش فضای حالتها در سیستم های هوشمند را نیز در اختیار می گذارد.

در انتها نمونه ای از یک سیستم پیاده سازی شده توسط این ابزار مورد بحث قرار می گیرد.

1- مقدمه

1-1- مقدمه‌ای بر سیستم‌های خبره و پایگاه دانش .

سیستم خبره عبارت است از یک سیستم نرم‌افزاری کامپیوتری حاوی دانش‌سازمان یافته چه‌از نوع دانش حقیقی (Actual) و چه از نوع دانش رهیافتی (Huristic) مربوط به یکی از محدوده‌های دانش انسانی که قابلیت استنتاج را بر اساس دانش ذکر شده داراست . [RUTH90] سیستم خبره برنامه‌ای کاربردی است که‌مسائل پیچیده‌ای را که حل‌آن نیاز به خبرگی انسانی دارد حل میکند . [ROLS88] سیستم خبره برنامه‌ای‌است‌که سعی در تقلید عملکرد و رقابت انسانهای خبره دارد . [SCHA90]

با عباراتی ساده‌تر یک سیستم خبره :

یک نرم‌افزار است . [RUTH90]

محتوی دانش انسانی ساخت یافته در سازمانی تحت عنوان پایگاه دانش است.چه‌دانش حقیقی وچه دانش‌رهیافتی . [RUTH90]
ساختمان خاصی برای نمایش دانش ذکر شده دارد . [SCHA90]
محتوی قواعد استنتاج برای‌تصمیم گیری ونتیجه گیری است . [RUTH90]
ارتباط روشنی بین قواعد استنتاج و شیوه نمایش دانش وجود دارد . [SCHA90]
دانش ذخیره شده و قواعد استنتاج محدود به زمینه خاصی از زمینه‌های خبرگی انسانی‌است . [RUTH90]
قابلیت توضیح دلائل نتیجه گیری و استنتاج های انجام شده را (معمولا) داراست . [RUTH90]
دارای واحد خاصی برای ایجاد ارتباط با کاربر است . [SCHA90] با توجه به مطالب گفته شده میتوان اجزاء اصلی یک سیستم خبره را با شکل زیر نشان داد :

شکل 1 - اجزاء اصلی سیستم‌های خبره . [CHAB89]

2-1- سیستم‌های خبره و پایگاه دانش پویا .

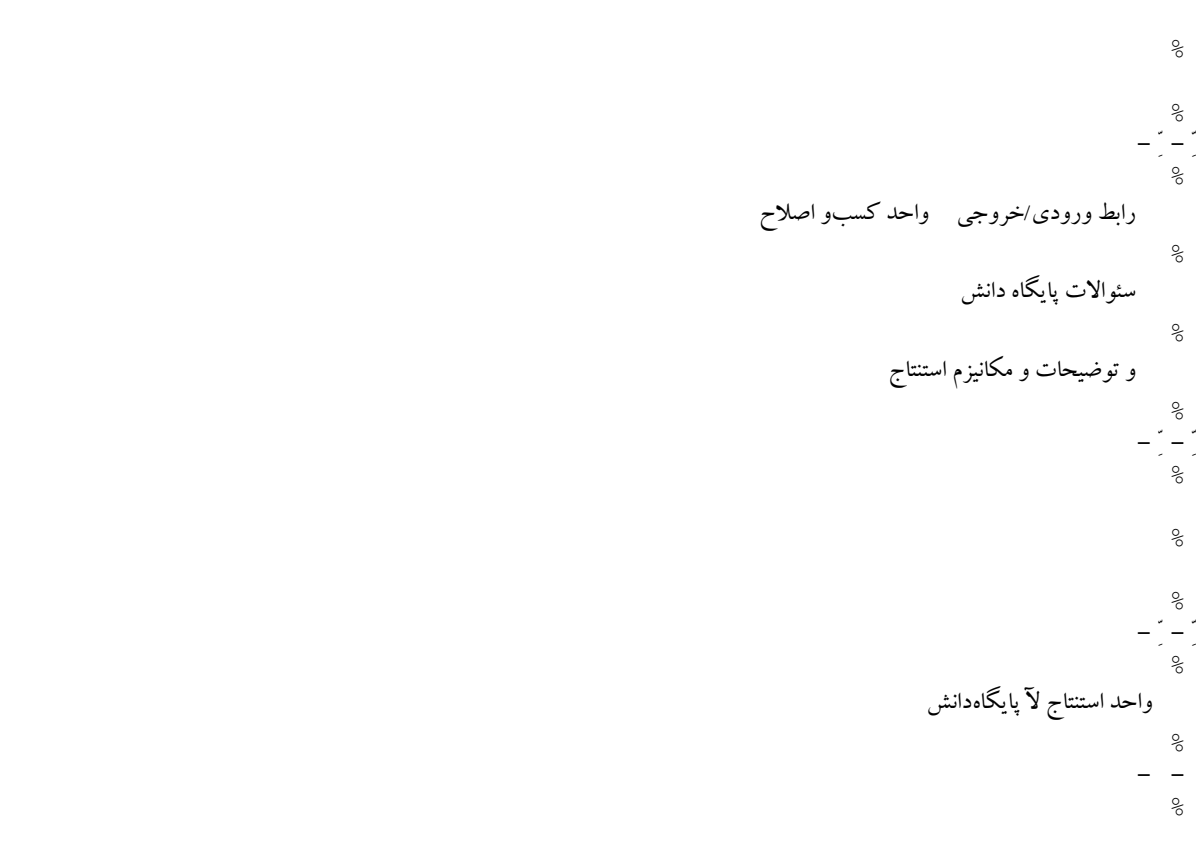
سیستم‌های خبره از نظر پویائی به دو وضعیت قابل تقسیم هستند :
1 - سیستم‌های خبره ایستا . (STATIC)
در اینگونه سیستم‌ها دانش و مکانیزم‌های استنتاج در هنگام طراحی و ساخت سیستم خبره تعریف‌شده و پس از ایجاد سیستم و شروع فعالیت‌آن امکان پذیر نمیباشد (مگر در محدوده بسیار کوچک) .
2 - سیستم‌های خبره پویا . (DYNAMIC)
در اینگونه سیستم‌ها دانش یا مکانیزم‌های استنتاج پس از ایجاد سیستم خبره در محدوده‌های معین قابل تغییر است .
البته باید در نظر داشت که با توجه به اینکه پویائی و ایستائی نسبی است بنابر این تعریف پویائی و ایستائی یک سیستم به طور مطلق امکان‌پذیر نیست و محدوده پویایی مشخص کننده این نسبیت است .

پویائی سیستم‌های خبره از جنبه دیگر نیز قابل بررسی است :

- 1 پویائی در پایگاه دانش .
- 2 پویائی در مکانیزم های استنتاج .

ایجاد سیستم‌های خبره پویا دشوارتر است و اینگونه سیستم‌ها پیچیده‌تر از سیستم‌های ایستا هستند . از طرفی ایجاد پویائی در مکانیزم‌های استنتاج بسیار پیچیده‌تر از پویائی در پایگاه دانش میباشد و گاه غیر ممکن مینماید . [RUTH90]

سیستم‌های خبره پویا علاوه بر دارا بودن تمام‌خصوصیات سیستم‌های خبره ایستا باید حاوی ساختمان نمایش دانشی باشند که قابلیت پویائی را امکان پذیر نماید . این سیستمها علاوه‌بر تمام قسمتهای یک سیستم خبره ایستا شامل قسمتی تحت عنوان " واحد کسب دانش (MODULE MODIFICATION/ KNOWLEDGE ACQUISITION) "نیز میباشند . [SCHA90] نمودار بعدی نشان دهنده اجزاءیک سیستم خبره پویا است :



شکل 2 - ساختمان عمومی یک سیستم خبره پویا . [SCHA90]

3-1- مراحل ودشواریهای پیاده سازی سیستم‌های خبره و پایگاه دانش .

در ایجاد یک سیستم خبره دانش مربوطه توسط خبره‌انسانی بصورت قواعد ، تعاریف و .... بیان شده و مهندس دانش ، دانش بیان شده را به همراه داده‌های موجود در پایگاه داده‌ها (احتمالا) به شیوه نمایش مناسب در پایگاه دانش ذخیره نموده‌و مکانیزم‌های استنتاج را ایجاد مینماید .

شکل 3 به صورت ساده نشان میدهد سازمان یک سیستم خبره و پایگاه دانش از دیدگاه ایجاد سیستم‌و استفاده ازآن چگونه‌است . از گامهائی که لازم است برای ایجاد سیستم خبره برداشته شود گامهای زیر اهمیت بسیار زیادی را دارا هستند [SCHA90] [ROLS88] :

کسب دانش مورد نظر .

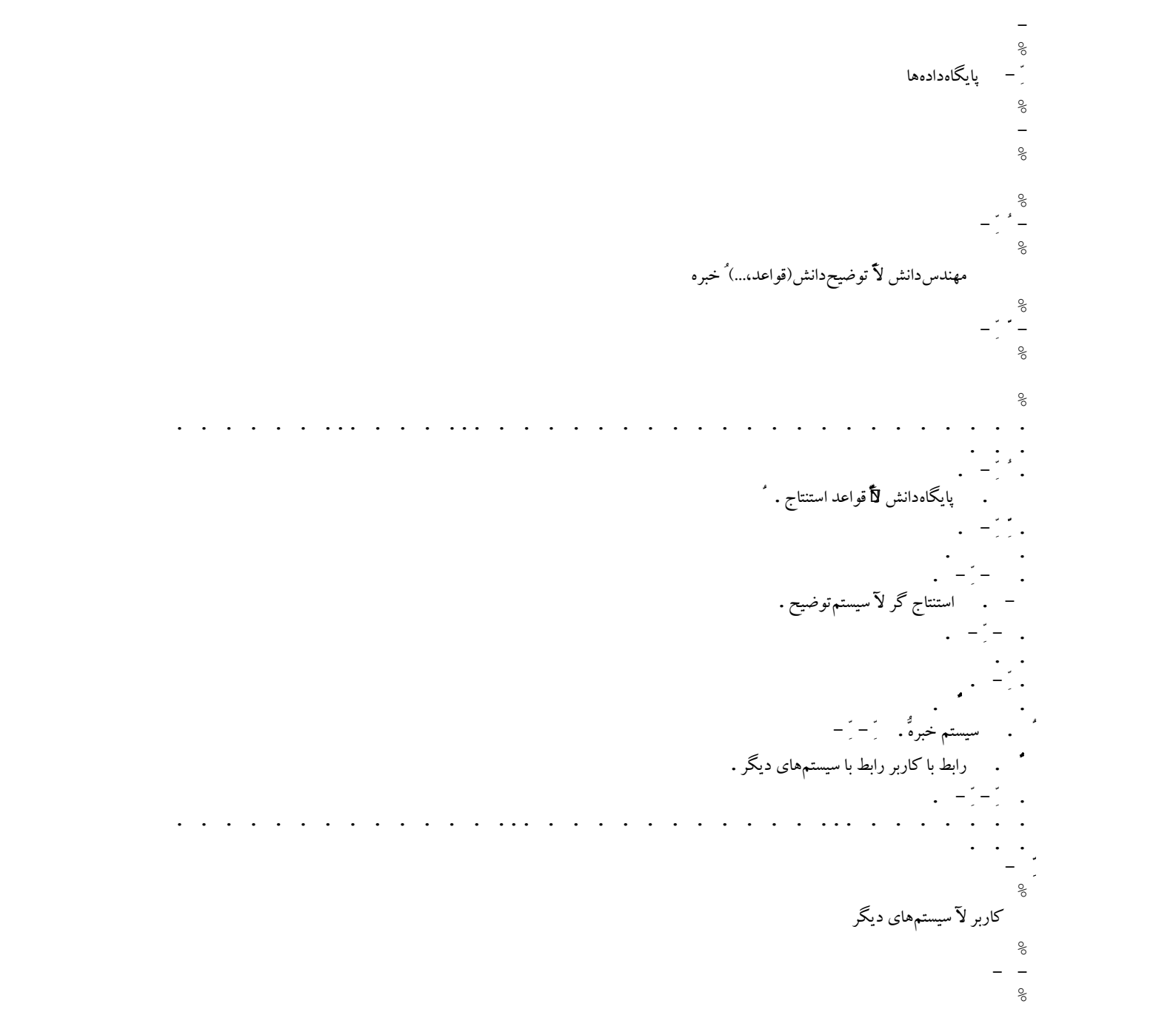
انتخاب شیوه نمایش دانش مناسب‌و شیوه‌مکانیزم‌استنتاج .

طراحی و ساخت روالهای ایجاد و استفاده پایگاه دانش با توجه به شیوه انتخاب شده .

طراحی و ساخت واحد استنتاج با توجه به شیوه انتخاب شده .

طراحی و ساخت رابط کاربرمناسب .

باید توجه داشت که در سه گام انتهائی ذکر شده قسمت اعظم هزینه و زمان صرف طراحی و پیاده سازی روالهائی میگردد که ارتباط چندانی با موضوع خبرگی و هوشمندی نداشته و صرفا عملیات سیستمی و معمولی در تمام سیستمها است. از این جمله اند عملیات پردازش فایل و ذخیره سازی پایگاه دانش و جستجو در اطلاعات‌موجود در پایگاه و ایجاد ارتباطات منطقی در سطح رکوردها بین اطلاعات موجود . انجام این عملیات‌نه تنها قسمت بسیار زیادی از هزینه و وقت طراحی و پیاده‌سازی را به خود معطوف



شکل 3 - سازمان چگونگی ایجاد واستفاده سیستم‌های خبره] بااصلاح، . [RUTH90

میکند بلکه ذهن طراحان را از تمرکز بر موضوع اصلی خبرگی (نظیر ارتباطات بین اجزاء و مکانیزم‌های کلان استنتاج) باز داشته و متوجه سطوح زیرین (نظیر عملیات در سطح فایل و رکورد) مینماید و از بهره‌وری متمرکز فکر طراحان جلوگیری میکند . این موضوع باعث میشود تا لزوم وجود ابزار مناسب برای رهائی از سطوح زیرین و ایجاد سطوح انتزاعی مناسب احساس گردد .

4-1- ابزار ایجاد سیستم خبره .

یک ابزار ایجاد پایگاه دانش یک بسته نرم‌افزاری است که ساخت یک پایگاه دانش کاربردی راآسان میکند . [RUTH90] این ابزار خود از ابزارهای دیگری تشکیل میگردند . این‌ابزارها به طراح کمک میکنند که ساختمان پایگاه دانش ایجاد شده توسط ابزار را به عنوان ساختمان مبنای کار خود قرار دهد و عملیات جستجو در پایگاه دانش و استنتاجات بر مبنای این جستجو و نیز ارائه‌توضیح به کاربر را توسط روالهای این ابزار انجام دهد و تنها به چگونگی عملیات کلان برای ایجاد سیستم خبره بیندیشد .

این ابزار ها معمولا بصورت زیر دسته بندی میگردند :

- 1 ابزار ایجاد و استفاده پایگاه دانش .
- 2 ابزار تعریف و استفاده مکانیزم‌های استنتاج و جستجو .
- 3 ابزار ایجاد زیر سیستم توضیح .
- 4 ابزار ایجاد و استفاده رابط کاربر .

ابزار چهارم بیشتر از ابزارهای دیگر در دسترس است و انواع مختلفی ازآن وجود دارد . به همین دلیل تلاش ما ایجاد سه ابزار دیگر است .

5-1- پارامترهای اساسی در ارزیابی ابزار ایجاد سیستم‌های خبره .

در ارزیابی یک ابزار ایجاد سیستم‌های خبره پارامترهای زیر

باید مورد بررسی قرار گیرند :

1-5-1- میزان نزدیک شدن استفاده کننده ابزار به‌اهدافی که درایجاد ابزار تعیین گردیده بود .

این اهداف به طور خلاصه‌عبارتند از :

- 1 انجام عملیات ایجاد و استفاده از پایگاه دانش به صورتی که کاربر تنها نوع وارتباطات‌بین موجودیت‌ها را تعیین نماید و از چگونگی ذخیره سازی و ارتباطات منطقی در سطح فایل مستقل باشد . [RUTH90]
- 2 انجام مکانیزم‌های جستجو و استنتاج به‌گونه‌ای که کاربر از چگونگی انجام عملیات مستقل باشد - 3 . [RUTH90]قابلیت تعریف پایگاه به صورت پویا (ایجادو حذف موجودیت‌ها و ارتباطات‌بین‌آنها در هنگام استفاده از پایگاه) .
- 4 قابلیت پویائی مکانیزم‌های استنتاج (ایجاد مکانیزم جدید در هنگام‌استفاده) .
- 5 ارائه توضیح به کاربر استفاده کننده سیستم خبره با توجه به مکانیزم‌های استنتاج موجود .
- 6 قابلیت پویائی سیستم توضیح (ایجاد عملگرهای توضیح برای مکانیزم‌های استنتاج ایجاد شده جدید) .

2-5-1- نرم‌افزار ایجاد شده توسط این ابزار معیارهای یک نرم‌افزار خوب را دارا باشد . از جمله معیارهای یک‌نرم‌افزار خوب (پارامترهای کیفیت نرم‌افزار)[PRES87] :

انعطاف‌پذیری Flexibelity

قابل‌آزمون بودن Testability

قابل حمل بودنPortability

قابلیت استفاده مجددReusability

قابلیت یادگیری سادهLearnability

قابلیت به‌کارگیری سادهUsability

کارائیEfficiency

قابلیت نگهداریMaintainability

قابلیت اتصال به‌سایر سیستم‌هاInterperability

3-5-1- ساده نمودن کار ایجاد سیستم خبره با توجه به پارامترهای زیر :

- 1 بر عهده گرفتن یا ساده کردن وظایف مهندس دانش با ارائه امکانات در زمینه‌های :

اکتساب دانش .

نمایش دانش .

استنتاج .

توضیح .

- 2 بر عهده گرفتن یا ساده کردن وظایف برنامه نویس با ارائه امکانات در زمینه‌های :

امکان ایجاد ارتباط با نرم‌افزارهای دیگر .

امکان عملیات پردازش متن دو زبانه .

. . . .

- 3 مجتمع بودن امکانات موردنیاز کاربر برنامه‌نویس . 4 - قابلیت استفاده در طیف وسیعی از محیط های عملیاتی و قابلیت استفاده مستقیم در سیستم‌های نرم‌افزاری مختلف .

2- ابزار ایجاد سیستم خبره مبتنی بر شبکه معنائی .

1-2- موضوع و هدف ابزار .

ابزار به عنوان وسیله ایجاد سیستم‌های خبره مبتنی بر شیوه نمایش دانش شبکه معنائی با قابلیت پویائی در ایجاد و توسعه پایگاه دانش و مکانیزم‌های استنتاج بصورت محدود مورد استفاده قرار میگیرد .

هدف از ایجاد ابزار :

سهولت در ایجاد سیستم خبره با فراهم‌آوردن مجموعه‌ای از امکانات با خصوصیات زیر :

قابلیت ایجاد و استفاده از پایگاه دانش به گونه‌ای که از شیوه نمایش دانش شبکه معنائی استفاده نماید .

دانش موجود در پایگاه دانش قابل تغییر و تعریف از نظر جنبه‌های زیر باشد :

- 1 حجم دانش .
- 2 ماهیت و نوع موجودیت‌ها .
- 3 ماهیت‌و نوع ارتباطات‌بین موجودیت‌ها .
- 4 محتوی موجودیت‌ها .
- 5 محتوی ارتباطات بین موجودیت‌ها .

مکانیزم‌های جستجوی متفاوتی موجود باشد :

- 1 جستجوی ساده (از طریق ارتباطات‌موجود) .
- 2 جستجوی توارثی بر مبنای توارث خصوصیات .
- 3 جستجوی توالی خاص (براساس یک‌توالی منطقی) 4 - جستجوی مرکب از دو حالت‌توالی‌خاص و توارث. مکانیزم‌های استنتاج با استفاده از مکانیزم‌های جستجو عمل نمایند .

قابلیت طراحی مکانیزم‌های استنتاج جدید بر اساس مکانیزم های جستجوی موجود ، با الگوربتمهای ساده امکان پذیر باشد .
قابلیت تعریف خصوصیات مربوط به جستجوها (توارث، توالی ،ارتباطات‌عادی و جهت‌دارنمودن ارتباطات) در پایگاه دانش .
قابلیت ارائه توضیح با دو نوع جمله بندی فارسی/لاتین .
قابلیت تعریف عملگرهای توضیح با توجه به نوع ارتباطات تعریف شده .
قابلیت اکتساب دانش و مکانیزم‌های جستجو در هنگام اجرا و عملیات عمومی سیستم .
قابلیت ترکیب و استفاده از شبکه معنائی توام با روشهای دیگر نمایش دانش) اختصاصا روش . (KN_PRESENTATION FRAME
قابلیت تعریف ، ایجاد ، استفاده‌و حذف یک پایگاه دانش و سیستم خبره از داخل یک برنامه دیگر مثلا اگر سیستم بنا به دلایلی به صورت موقت (TEMPORARY) ایجاد شده باشد .
قابل استفاده بودن ابزار در یک زبان برنامه سازی و در تمام سیستم‌های نرم‌افزاری که توسط کامپایلر زبان برنامه سازی ایجاد میگردند .

2-2- محیط عملیاتی و محدوده کار ابزار .

سیستم به عنوان یک UNIT در برنامه‌های زبان پاسکال تحت کامپایلر TURBO PASCAL قابل استفاده است .
ابزار در دو محدوده عمل میکند :
الف - ساختمان داده‌های محدود .
در این حالت حداکثر هزار موجودیت قابل تعریف در سیستم است . در این حالت از ابزار SEM_NET.TPU استفاده میگردد .
ب - ساختمان داده‌های نامحدود .
در این حالت بر تعداد موجودیت‌ها محدودیتی وجود ندارد . در این حالت از ابزار SEM_NETL.TPU استفاده میگردد .
در هر یک از دو حالت محدودیتی از نظر تعداد ارتباطات وجود ندارد . در حالت اول‌سیستم سریعتر عمل میکند .
ابزار از نظر نوع رکوردهای اطلاعاتی منسوب به دانش ذخیره شده محدودیتی ندارد و اندازه این رکوردها نیز میتواند متغیر باشد .

3-2- منطق و سازمان پایگاه دانش در ابزار .

سازمان پایگاه دانش بر اساس شیوه نمایش دانش شبکه معنائی شکل گرفته است .

1-3-2- شبکه معنائی .

یک شبکه معنائی بر نمایش گراف شکل ارتباطات بین اجزاء یک محدوده معین بنا شده است . [ROLS88] اجزاء اساسی یک گراف را گره‌ها و لبه ها تشکیل میدهند . [ROBN85] گره‌ها نمایانگر موجودیت‌ها و لبه ها نمایانگر ارتباطات بین موجودیت‌ها میباشند . [ROLS88]

2-3-2- موجودیت‌ها .

در این پایگاه هر موجودیت میتواند هر ساختمان داده ممکن در زبان پاسکال باشد و ساختار حتی میتواند از نظر اندازه پویا باشد . در واقع تنهاآدرس ابتدای هر ساختمان داده در فایل داده‌ها به عنوان موجودیت مورد استفاده قرار میگیرد .

3-3-2- ارتباطات‌بین موجودیت‌ها .

در این پایگاه دانش هر موجودیت میتواند با موجودیت‌های دیگر ارتباط داشته باشد . این ارتباط میتواند یک‌طرفه یا دوطرفه باشد . ارتباطات وزن دار هستند یعنی هر ارتباط با توجه به وزن‌آن ارتباط بیانگر مفهوم تعریف شده‌ای است . [ROB85]

4-3-2- نوع ارتباط ها .

هر ارتباط باید از نوع خاصی باشد . یک نوع لبه نشانگر خصوصیاتی است که لبه های ازآن نوع دارای‌آن هستند . مثلا یک نوع لبه میتواند " فرزند " باشد . از طریق این لبه هر شخص میتواند با ولی خود ارتباط داشته باشد . مثلا اگر موجودیت "علی" با "کریم" رابطه "فرزند" داشته باشد

به معنای آن است که علی فرزند کریم است و کریم ولی علی است . مشخص است که رابطه "فرزند" یکطرفه است. ولی مثلا رابطه "دوست" دوطرفه است. در این ابزار در پایگاه دانش میتوان 225 نوع لبه را تعریف نمود هر نوع لبه همچنین میتواند در هنگام عملیات سیستم ایجاد و یا حذف گردد.

5-3-2-توارث خصوصیات .

یکی از مهمترین خصوصیات شبکه معنائی توارث خصوصیات است . هر لبه میتواند به عنوان عامل انتقال خصوصیات از یک گره به گره دیگر بر طبق قانون توارث خصوصیات باشد . بدین ترتیب که خصوصیاتی که در یک گره وجود دارند از طریق لبه‌های خاص میتوانند به گره‌های دیگر به ارث برسند . لبه‌هائی که خصوصیات یک موجودیت را به ارث میرسانند باید از انواع لبه هائی باشند که در هنگام تعریف، توارثی تعریف گردیده‌اند . در واقع یکی از خصوصیات یک نوع لبه توارثی بودن یا نبودن‌آن نوع لبه‌است. همچنین یک لبه از نوع توارثی میتواند عامل ارث در جهت مستقیم و یا معکوس و یا هردو طرف باشد . به عنوان مثال در شکل زیر یک قسمت از یک پایگاه دانش نمایش داده شده است. هر موجودیت با یک دایره و هر ارتباط با یک خط مشخص شده است. در این شکل دو نوع ارتباط وجود دارد . 1 - ارتباط "هست‌یک" از نوع لبه توارثی با جهت ارث معکوس و 2 - ارتباط "داردیک" . هنگامی که در مورد "سگ" سئوال "چه چیز دارد؟" مطرح میشود علاوه بر "دم" که به طور مستقیم با"سگ" در ارتباط است خصوصیات یک "حیوان" که دارای سر و پا نیز هست به "سگ" ارث میرسد و جواب این سئوال عبارت خواهد بود از : دم،سر،پا

پا

سر

داردیک

داردیک

دم حیوان

داردیک هست‌یک

سگ

شکل 4 - توارث خصوصیات حیوان به سگ

6-3-2- توالی ارتباطات .

یک ارتباط میتواند به صورت مجازی از تعریف توالی چند ارتباط دیگر تعریف گردد و مانند ارتباطهای دیگر مورد پردازش قرار گیرد . بعنوان مثال در یک پایگاه دانش اطلاعات خانواده ارتباط خاله بودن بدینصورت تعریف میشود :

خاله = -<مونث<->فرزند>-<فرزند<-<مونث<-<فرزند>

این توالی بیانگرآن است که " خاله " شخص مونثی است که فرزند شخصی است که فرزند مونثی دارد که او نیز فرزندی دارد .

در شکل 5 نمودار این ارتباط نشان داده شده‌است .

4-2- منطق و مکانیزم‌های جستجو و استنتاج در ابزار .

پایه مکانیزم‌های استنتاج در این‌ابزار بر مکانیزم های جستجو استوار است .

-2-4-1 مکانیزم‌های جستجو

ابزار مکانیزم‌های جستجوی زیر را در اختیار قرار میدهد .

-2-4-1 جستجوی ساده .

در این روش کلیه گره‌های مرتبط به گره مبداء مورد بررسی و در صورت تطابق با شرایط خواسته شده به عنوان نتیجه جستجو اعلان میگردند .

X

فرزند فرزند

مونثY A

مونث

فرزند

B

شکل 5 - زنجیره توالی ارتباط خاله بودن Y . خاله B است .

-2-4-2 جستجوی توارثی .

در این روش کلیه گره‌های مرتبط به گره مبداء مورد بررسی و در صورت تطابق با شرایط خواسته شده به عنوان نتیجه جستجو اعلان میگردند در صورتی که لبه‌ای از نوع توارثی باشد جستجو به طرف دیگر لبه نیز منتقل میشود و این عمل به صورت بازگشتی بر روی تمام لبه های توارثی مرتبط دیگر نیز تکرار میگردد(به‌مثال بخش5-3-2 توجه‌کنید). 2-4-2-3- جستجو بر اساس توالی منطقی ارتباطات .

در این حالت جستجو بر اساس زنجیره توالی تعریف شده انجام میگیرد . به عنوان مثال در مثال بخش 6-3-2 از طریق این نوع جستجو میتوان خاله یک‌فرد را یافت‌در حالیکه تنها ارتباط "مونث" و "فرزند" به طور حقیقی در پایگاه تعریف شده است .

-2-4-4 جستجوی ترکیبی از توارث و توالی .

دراین حالت ارتباطات مجازی توالی منطقی بر اساس جستجوی توارثی انجام میگیرد . مثلا در یک پایگاه دانش مربوط به کارمندان اداره میتوان لیست خاله تمام کارمندانی که عضو بخش مالی یا زیر شعبه‌های‌آن هستند تنها با یک دستورالعمل یافت . در صورتی که رابطه "کارمند" و "شعبه" خصوصیت توارث مستقیم را دارا باشند .

با توجه به شکل 6 جواب زهره و مریم خواهد بود .

-2-4-3 مکانیزم‌های استنتاج

با توجه به مکانیزم‌های جستجو مکانیزم های استنتاج به سادگی قابل طراحی هستند . نمونه مکانیزم‌های استنتاج در بخش 7-2 ارائه گردیده است .

فاطمه حسین

فرزند فرزند فرزند فرزند

مونث بطول مریم مونث مونث زهرا زهره مونث

فرزند فرزند

رضا علی

کارمند کارمند

حسابداری صندوق

شعبه شعبه

بخش مالی

شکل 6 - یافتن لیست خاله کارمندان بخش‌مالی

5-2- منطق و مکانیزم سیستم توضیح .

زیر سیستم توضیح توسط واحد استنتاج فراخوانی میشود و جملاتی را با توجه به مکانیزم استنتاج و عملیات انجام شده در فایلی ذخیره میکند . با درخواست کاربر محتویات این فایل در اختیار کاربر برای مشاهده توضیحات قرار میگیرد و پاک میشود .

1-5-2- مکانیزم ساخت جملات ساده .

مکانیزم‌ساخت‌جملات توضیح براساس‌عبارات‌ربطی (عملگر ارتباط) که‌در هنگام تعریف ارتباطات به سیستم معرفی میگردند انجام میشود . هر ارتباط در دو وضعیت مستقیم و معکوس دارای عبارات ربطی میباشد که ارتباط بین دو موجودیت را در قالب یک جمله بیان میکند . به عنوان مثال در مورد نوع ارتباط (لبه) "فرزند" دو عبارت "فرزندی دارد به نام" و "فرزند کسی است‌به نام" به عنوان عبارات ربط بین موجودیت ها بترتیب در حالت معکوس و مستقیم مورد استفاده قرار میگیرند و بدین سان مثلا ارتباط (علی ، فرزند ، حسین) بصورت دو جمله زیر توسط زیر سیستم توضیح تعبیر میگردد :

در حالت مستقیم - " علی فرزند کسی است به نام حسین "

در حالت معکوس - " حسین فرزندی دارد به نام علی "

2-5-2- مکانیزم ساخت جملات براساس توارث خصوصیات .

در مواردی که بر اثر اصل توارث خصوصیات ، خصوصیات یک موجودیت به موجودیت دیگر به ارث میرسد سیستم توضیح انتقال خصوصیات را با چنین جمله ای بیان میکند :

" چون سگ هست‌یک حیوان پس " :

و پس ازآن ارتباطات مربوط به حیوان را که به سگ به ارث میرسد از طریق جملات ساده بیان میکند .

3-5-2- مکانیزم ساخت جملات بر اساس زنجیره توالی .

در این گونه موارد سیستم ابتدا زنجیره توالی را توضیح میدهد و سپس دلیل انتخاب ارتباط را با توضیح پیمایش زنجیره توالی از طریق جملات ساده بیان میکند .

در مثال بخش 4-2-4-2 با دستور یافتن نام‌خاله تمام کارمندان بخش مالی توضیحات شکل صفحه بعد ارائه میشود .

> توصیف یک خاله > " یک‌خاله "
" زنی است "
" که فرزندکسی است به نام " A
" که فرزندی دارد به نام " B
" که زنی است "
" که فرزندی دارد به نام " C
> انتهای توصیف یک‌خاله> " حال " :
> توارث از شعبه > " چون بخش مالی شعبه ای دارد تحت عنوان حسابداری "
> توارث کارمند > " چون حسابداری کارمندی دارد به نام رضا "
> اعلام‌نام خاله رضا > " چون رضا خاله ای دارد به نام مریم "
> دلیل خاله‌رضا بودن > " چون رضا فرزند کسی است به نام بطول "
" چون بطول زن است "
" چون بطول فرزند کسی است به نام فاطمه "
" چون فاطمه فرزندی دارد به نام مریم "
" چون مریم زن است "
> توارث از شعبه > " چون بخش مالی شعبه ای دارد تحت عنوان صندوق "
> توارث از کارمند > " چون صندوق کارمندی دارد به نام علی "
> اعلام‌نام‌خاله علی > " چون علی خاله ای دارد به نام زهره "

> دلیل‌خاله‌علی‌بودن > " چون علی فرزند کسی است به نام زهرا "
" چون زهرا زن است "
" چون زهرا فرزند کسی است به نام حسین "
" چون حسین فرزندی دارد به نام زهره "
" چون زهره زن است "

شکل 7 - توضیحات جستجو برای یافت نام‌خاله کارمندان بخش‌مالی .

6-2- سازمان ابزار و ارتباطات‌اجزاء .

ابزار از سه قسمت اساسی زیر تشکیل میگردد

1-6-2- بخش ایجاد و استفاده از پایگاه دانش .

این بخش به عنوان هسته اصلی بخشهای دیگر و واحد مرکزی ابزار عمل میکند .

2-6-2- بخش توضیح .

کنترل این بخش توسط بخش استنتاج انجام میشود .

3-6-2- بخش استنتاج .

عملیات جستجو در پایگاه دانش و استنتاج توسط این بخش انجام میگردد .
شمای تصویری این سه بخش‌و ارتباطات‌بین این اجزاء در شکل بعدی نشان داده شده است .

سیستم نرم‌افزاری میزبانُ

پرسش پاسخ توضیح دانش

سیستم توضیح

شکل 8 - سازمان ابزار و ارتباطات‌اجزاء

شرح مختصری از روالهای قابل استفاده توسط ابزار به همراه یک برنامه نمونه در ضمیمه 3 ذکر گردیده است .

3- بررسی ابزار ایجاد سیستم‌خبره مبتنی بر شبکه معنائی .

در این بخش به بررسی ابزار ایجاد سیستم خبره مبتنی بر شبکه معنائی میپردازیم .

3-1 سیستمهائی که بوسیله این ابزار پیاده میشوند .

هر سیستم خبره که دانش مربوط به‌آن قابل نمایش توسط شبکه معنائی باشد قابل پیاده سازی توسط این ابزار است . با توجه به این موضوع تعداد زیادی از مسائل و سیستم هامیتوانند تحت پوشش این ابزار قرار گیرند زیرا بسیاری از مسائل کاربردی ماهیت شبکه‌ای و گراف شکل دارند . ازآن جمله میتوان به‌موارد زیر اشاره نمود :

قطعات مورد نیاز برای ساخت یک محصول در یک کارخانه تولیدی در یک سیستم انبارداری .
تمام ادارات ، قسمتها ، کارمندان ، عمل‌ها در یک سیستم اداری و یا صنعتی .
مسیرها در مسئله ترابری .
پیمایش گراف تشخیص (نمونه در بخش 2-7-3) .
کلیه ساختارهای درختی .
پیمایش فضای حالتهای گراف شکل .

سیستم‌های مترجم زبانهای انسانی .

بطور کلی با توجه به اینکه میتوان اغلب روشهای نمایش دانش را به گونه ای به شبکه معنائی تبدیل نمود) از جمله PRODUCTION SYSTEMSو FRAME و (SCRIPT این ابزار را میتوان پوشا بر اغلب مسائل سیستم‌های خبره دانست .

2-3- پایگاه دانش .

1-2-3- شیوه نمایش دانش .

با توجه به مطالب بیان شده در بندهای قبلی علاوه بر شیوه نمایش مبنای شبکه معنائی دو شیوه نمایش دانش FRAME و SCRIPT را به صورت مستقیم و شیوه نمایش دانش SYSTEMS - PRODUCTION را به صورت ضمنی میتوان در این پایگاه نمایش داد .

2-2-3- حجم دانش .

حجم دانش ذخیره شده در سیستم SEM_NET از لحاظ تعداد موجودیت محدود به 1024 موجودیت، ولی سیستم SEM_NETL محدودیتی از نظر تعداد موجودیت ندارد . هیچیک از سیستم‌های ذکر شده از نظر تعداد ارتباطات محدودیتی ندارند اما ایجاد تعداد زیادی ارتباط (بیش از 100) بر روی یک موجودیت باعث افت کارائی میشود و در اینگونه موارد استفاده از روشهای مناسب در تعریف موجودیت ها و نوع ارتباطات میتواند از پیش‌آمدن این موضوع جلوگیری نماید .

حجم تعداد نوع ارتباط محدود به عدد 225 است .

3-2-3- پویائی دانش .

دانش موجود در پایگاه دانش از نظرات زیر پویا است :

1- پویائی از نظر تعداد موجودیت‌ها(ایجاد-حذف) .
2- پویائی از نظر تعداد نوع ارتباطات(ایجاد) .
3- پویائی از نظر تعداد ارتباطات(ایجاد-حذف) .
4- پویائی از نظر محتوای موجودیت‌ها) شکل و اندازه رکورد و ساختمان موجودیت مثلا FRAME یا . (SCRIPT
5- پویائی از نظر محتوای ارتباطات(وزن ارتباطات) .
6- پویائی از نظر خصوصیات نوع ارتباط (توارثی - جهت) .
7- پویائی از نظر توالی منطقی زنجیره توالی (ایجاد توالی). 8- پویائی از نظر ماهیت و نوع ارتباطات بین موجودیت‌ها .

3-3- مکانیزم‌های جستجو .

1-3-3- انعطاف‌پذیری مکانیزم‌ها .

انعطاف پذیری مکانیزم‌ها از دو جنبه قابل بررسی است .

1-1-3-3- انعطاف پذیری مکانیزم‌های جستجو .

پارامترهای جستجو انعطاف‌پذیری جستجو را افزایش میدهند . پارامترهائی که در جستجو نقش دارند عبارتند از :

موجودیت‌آغازین .
نوع ارتباطات .
وزن ارتباطات .
جهت ارتباطات .
جهت توارث .

جهت توالی .

-3-3-1-2 قابلیت طراحی واحد استنتاج .

واحد استنتاج با تعداد دستورات بسیار کم‌و با ساختمان نسبتا ساده توسط استفاده کننده قابل طراحی است . نمونه واحد استنتاج در واحد استنتاج برنامه عیب یاب اتومبیل قابل مشاهده است .

-3-3-2 تعداد مکانیزم‌ها

مکانیزم‌های جستجوی زیر قابل استفاده اند .

ساده

توارثی

توالی

مرکب(عمومی)

-3-3-3 ایجاد مکانیزم‌های جدید توسط کاربر .

با در اختیار قرار گرفتن زیر روالهای یافت گره جاری ، یافت اولین ارتباط و یافت ارتباط بعدی که در روالهای مربوط به پایگاه‌دانش موجود است ، استفاده کننده میتواند روالهای جستجوی مورد نیاز خود را خود طراحی نماید .

-3-4 سیستم توضیح .

-3-4-1 پویائی سیستم توضیح .

وجود قابلیت تعریف و تغییر عبارات توصیفی موجب ایجاد پویائی سیستم توضیح نسبت به تغییر عبارات (در اصلاح عبارات) و تغییر نوع و ماهیت ارتباطات (در اصلاح یا تعریف نوع ارتباطات) است .

-3-4-2 مکانیزم ساخت جملات .

مکانیزم ساخت جملات با توجه به‌آنکه تنها از ترکیب نام موجودیت اول به علاوه عبارت ربط و موجودیت دوم تشکیل میگردد مکانیزم چندان قوی به‌نظر نمیرسد . اما سادگی مکانیزم به‌آن منجر میشود که تعریف کننده پایگاه دانش به سادگی بتواند چگونگی جمله و عبارات ربطی مربوط به ارتباط مورد نظر را برای سیستم تنها با یک عبارت‌در حالت مستقیم و یک عبارت در حالت معکوس تعریف نماید . هر چند جملات ساخته شده در این عبارت چندان دلچسب نیستند اما معنای مورد نظر را میرسانند .

-3-4-3 جمله سازی فارسی در سیستم توضیح .

سیستم این امکان را میدهد تا جملات در دو وضعیت فارسی و یا لاتین شکل بگیرد و این کار با تعریف یک متغیر در ابتدای کار به یکی از دو صورت زیر امکان پذیر است :

```
; FARSI =: EXPLAIN_TEXT_FARSI_LATIN
```

و یا :

```
; LATIN =: EXPLAIN_TEXT_FARSI_LATIN
```

این متغیر ماهیت جمله بندی و نیز کلماتی که در جمله بندی توسط سیستم مورد استفاده قرار میگیرند (نظیر چون ، پس ، BECAUSE، SO) را تغییر میدهد و بنابراین از سیستم توضیح هم در محیط فارسی و عربی و هم در محیط لاتین میتوان استفاده نمود .

-3-5 بررسی‌ابزار از نظر پارامترهای کیفیت نرم‌افزار .

انعطاف پذیریFLEXIBLITY

انعطاف پذیری از جنبه‌های زیر قابل بررسی است(که توضیحات آن در بندهای گذشته ارائه گردید) :

انعطاف‌پذیری پایگاه دانش .

انعطاف‌پذیری مکانیزم‌های استنتاج .

انعطاف‌پذیری سیستم توضیح .

قابلیت آزمون TESTABILITY

وجود توابعی در سیستم که مقادیر و وضعیت‌موجود پایگاه‌دانش راباز میگردانند در(مثلابازگرداندن وضعیت نوع لبه) و نیزوجود مکانیزم یافت خطا درسیستم با متغیر SN_ES_ERROR و تعیین شدن این متغییر در تمام عملیات داخلی ابزار و تابعی که متن خطای مربوطه را به فارسی یا لاتین بازمیگرداند به‌آزمون نرم‌افزار کمک میکند .

قابلیت حمل PORTABILITY

با وجود فایل OBJ ابزار این ابزار قابل استفاده بر روی تمام نسخه‌های (VERSIONS) کامپایلر TURBO PASCAL-BORLAND میباشد .

قابلیت استفاده مجدد REUSABILITY

اصولا ماهیت یک ابزار مشخص کننده این قابلیت برای استفاده در سیستم‌های نرم‌افزاری مختلف از یک نرم‌افزار مشخص است .

قابلیت یادگیری LEARNABILITY

محدود بودن تعداد دستورات و نیز راهنمای استفاده و مستندات استفاده سیستم و همچنین ساده بودن طریقه استفاده و وجود چند برنامه نمونه ضریب یادگیری کار با ابزار را افزایش میدهد .

قابلیت استفاده USABILITY

قابل استفاده بودن سیستم در اکثر موارد مطرح شده بنابر مندرجات بند 1-3 موید این خصوصیت میباشد .

کارائی EFFICIENCY

باآزمایشاتی که بر میزان سرعت و حجم حافظه مصرفی سیستم انجام گردیده سرعت عملیات از سیستمهای سنتی موجود بسیار سریعتر و حجم حافظه مصرفی اندکی افزونگی دارد که با توجه به وضعیت عمومی سیستم قابل چشم پوشی است .(محاسبه دقیق کارائی یک ابزار بنا بر حجم بالای عملیات و بنا به ماهیت این مقاله در این مقال نمیگنجد) .

قابلیت نگهداری سیستم MAINTAINABILITY .

بنا به سادگی روالهای مورد استفاده و مشخص بودن وضعیت و ساختمان عمومی عملیات داخلی در مستندات سیستم امکان نگهداری سیستم و توسعه سیستم بر روی ابزار وجود دارد .

قابلیت اتصال به نرم افزار های دیگر INTERPERABILITY .

قابلیت استفاده این ابزار در یک زبان برنامه نویسی این امکان را فراهم می‌آورد تا این سیستم بتواند با هر سیستم دیگری که توسط این زبان ایجاد شده یا از طریق کامپایلر امکان ایجاد ارتباط باآن وجود دارد ارتباط ایجاد نماید و به سیستمهای دیگر متصل شود .

6-3 کاربرد ابزار در پیمایش فضای حالتهادر سیستم‌های هوشمند .

یکی از متداول‌ترین روشها در پیاده‌سازی سیستم‌های هوشمند ایجاد فضای حالتها و پیمایش این فضا از یک حالت‌آغازین به یک حالت پایانی طی شرایط خاصی است. در صورتیکه در فضای حالتها هر حالت بعنوان یک موجودیت (گره) تعیین گردد و امکان حرکت از یک‌حالت به حالت دیگر با یک ارتباط (لبه) تعریف شود ، باایجاد قوانین منطقی مناسب در انتخاب موجودیت بعدی از میان موجودیت های مرتبط با موجودیت جاری به سادگی میتوان با طراحی یک واحد استنتاج بر مبنای این قوانین منطقی با استفاده از امکانات این ابزار در جستجوی ارتباطات وقابل تعریف بودن نوع ارتباطات ، با توجه به شرایط جاری لبه با نوع و وزن مناسب‌را انتخاب نمود و موجودیت بعدی (حالت جدید) را به عنوان موجودیت جاری در نظر گرفت (پیمایش فضا) .

نمونه‌ای از پیمایش فضای حالتها در یک سیستم عیب یاب سیستم استارت اتومبیل نشان‌داده شده است .

این برنامه از تعدادی حالت تشکیل میگردد که هر حالت بیانگر موقعیت خاصی از وضعیت امکان پذیر در جستجوی پیرامون عیب سیستم استارت اتومبیل است . برنامه از یک حالت ابتدائی که کاربرآنرا اعلام میکندآغاز میشود و تا رسیدن به یک حالت جواب ادامه پیدا میکند . متن این برنامه و توضیحات مربوط به‌آن در ضمیمه 1 وجود دارد .

4- بررسی یک سیستم پیاده سازی شده توسط ابزار سیستم‌سمپاد 1 .

1-4- هدف از ایجاد سیستم سمپاد 1 .

سیستم سمپاد 1 (سیستم محاوره‌ای پایگاه دانش) صرفا به عنوان وسیله کمک‌آموزشی درآشنائی و ملموس شدن ماهیت سیستم‌های خبره و هوشمند برای برنامه نویسان سنتی که میخواهند با روشهای هوش مصنوعی‌آشنا گردند ایجاد گردید . این امر از دو طریق انجام میشود :

- 1 سیستم سمپاد 1 کلیه عملیات ایجاد پایگاه دانش و استفاده از روالهای عملیات سیستم خبره را با محاوراتی با کاربر انجام میدهد . این محاورات به گونه‌ای انجام میشود که کاربر با ماهیت عملیات انجام شده توسط سیستم‌آشنا میگردد .
- 2 متن این برنامه (که تنها در حدود 200 خط برنامه است) در اختیار علاقمندان قرار میگیرد تا هم با چگونگی استفاده از ابزار ایجاد سیستم‌های خبره‌آشنا شوند و هم مکانیزم عملیات یک برنامه هوش مصنوعی را مشاهده نمایند .

2-4- چگونگی عملکرد سیستم سمپاد 1 .

سیستم با انجام محاوراتی با کاربر ایجاد یک پایگاه دانش و استفاده ازآنرا توسط کاربر انجام میدهد . سیستم تنها یک برنامه واسط است که استفاده از روالهای ابزار را مستقیما به کاربر محول میکند . در محاورات از چنین جملاتی استفاده میگردد :

موجودیت‌جدید ایجادکنید
نوع‌ارتباط‌جدید ایجادکنید
وضعیت‌ارتباط راچاپ‌کنید
وضعیت‌ارتباط راتعیین‌کنید
وضعیت‌توضیح‌ارتباط راچاپ‌کنید
وضعیت‌توضیح‌ارتباط راتعیین‌کنید
ارتباط جدیدایجادکنید
ارتباط راحذف‌کنید
موجودیت راحذف‌کنید
دستگاه‌خروجی راتعیین‌کنید
جستجوکنید
توضیح‌دهید
ارتباط مرکب‌ایجادکنید

نمونه‌ای از محاورات انجام شده با سیستم در ضمیمه 2 وجود دارد .

3-4- ساختمان عمومی سیستم سمپاد 1 و استفاده از ابزار ایجاد پایگاه دانش پویا .

سیستم از دو قسمت تشکیل میگردد :

1-3-4- بخش رابط کاربر

وظیفه این بخش انجام محاورات با کاربر است. روالهای این بخش ماهیت سنتی دارند و از ساده‌ترین الگوریتمهای ممکن برای سادگی یادگیری و جلوگیری از پیچیدگی تشکیل میشوند .

2-3-4- بخش کنترل و اعمال خواسته‌های کاربر

این بخش که در واقع بخش اصلی سیستم است وظیفه ترجمه خواسته های کاربر را به فراخوانی مناسب روالهای ابزار انجام میدهد . عملیات انجام شده به گونه‌ایست که دستوراتی را که کاربر صادر میکند با دستوراتی که برنامه به ابزار میدهد تناظر یک به یک دارد . از این رو ماهیت چگونگی انجام عملیات توسط ابزار برای کاربر سمپاد 1 روشن است .

5- نگاهی به ایجاد ابزارهای دیگر .

پیشرفت کاربرد سیستم‌های خبره در موارد حقیقی روزمره و نفوذ روشهای هوشمند در ایجاد سیستمهای واقعی در سالهای اخیر بصورت چشمگیری ملاحظه میشود . زبانهای برنامه سازی مناسب برای هوش مصنوعی و سیستم‌های خبره نظیر PROLOG و LISP هنوز قابلیتها و ابزارهای محیطی (ENVIRONOMENT TOOLS) مناسب برای ایجاد سیستم‌های بزرگ و کاربردی را نیافته‌اند . در حالیکه زبانهای برنامه‌سازی نظیر TURBO PASCAL و C++ با وجود امکانات جنبی و محیطی مناسب بطور عادی برای پیاده سازی مکانیزم‌های هوشمندانه مناسب نیستند . اینگونه مکانیزم‌ها برای پیاده سازی در این زبانها تبدیل به عملیات بسیار پیچیده و پرحجمی در سطوح برنامه‌نویسی سنتی میگردند . به همین دلیل اغلب در برنامه نویسی سیستم‌های بزرگ برنامه‌نویس از خیر استفاده از روشهای هوش مصنوعی میگذرد . وجود ابزارهائی نظیر ابزاری که در این مقاله برآن سخن رفت امکان استفاده از روشهای هوش مصنوعی را در این زبانها ممکن و ساده میسازد . استعداد نیاز به این ابزارها در زمینه‌های مختلف هوش مصنوعی گسترده است . ولی ساخت این ابزار ها میسر نیست مگر به همت دانشگاهیان در این امر . . . . . .

6- نتیجه‌گیری .

با توجه به‌آنچه گفته‌شد ، این‌ابزار به عنوان وسیله‌ایجاد سیستم‌های خبره مبتنی بر شیوه نمایش دانش شبکه معنائی با قابلیت پویائی در ایجاد و توسعه پایگاه دانش و مکانیزم‌های استنتاج بصورت محدود مورد استفاده قرار میگیرد .

این ابزار برای پیاده سازی هر سیستم خبره که دانش مربوط به آن قابل نمایش توسط شبکه معنائی بوده و ماهیت شبکه‌ای و گراف شکل یا درختی دارد مناسب است . با توجه به اینکه میتوان اغلب روشهای نمایش دانش را به گونه ای به شبکه معنائی تبدیل نمود این ابزار را میتوان برای اغلب مسائل سیستم‌های خبره بکار گرفت .

قدردانی .

پس از شکر گزاری خداوند متعال بر عنایت در تهیه این مقاله لازم است تا از خانمها مریم مجیدی و افسون قائمی که در ویرایش متن مقاله همکاری نموده‌اند قدردانی گردد .

مراجع .

ARTIFICAL INTELLIGENCE AND" . David W.Rolston : [ROLS88] . 1988. McGRAW-HILL . " SYSTEMS DEVELOPMENT INTEGRATING KNOWLEDGE-BASED" . Ruth Kerry : [RUTH90] EXPERT ELLIS HORWOOD 1990 . " MANAGMENT SYSTMES THE RELIABILITY OF " . Erik Hollnagel : [ERIK89]

. ELLIS HORWOOD 1989
Christopher : [CHAB89] . " EXPERT SYSTEMS . AND DATABASE .PASCAL

. TURBO GALGOTIA 1989 & ARTIFICAL INTELLIGENCE " . F.Chabris
. " ARTIFICIAL INTELLIGENCE " . Robert J.Shalkoff : [SCHA90] [HERB90]
ARTIFICIAL " . Herbert Schildt :
McGRAW-HILL 1990
Roger : [PRES87]
" . S.Pressman . . " INTELIGENCE USING C . McGRAW-HILL 1990 .
McGRAW_HILL 1987 : [BELL87]
SOFTWARE " . BELL-MORREY-PUGH . " SOFTWARE ENGINEERING . Prentice/Hall
1987 " . Robin J.Wilson : [ROBN85]

. " INTRODOCTION TO GRAPH THEORY . " ENGINEERING
. TECHNICAL 1985 & LONGMAN SCIENTIFIC

ضمائم

> پایان‌صفحه<

ضمیمه 1 - متن‌برنامه عیب یاب سیستم استارت اتومبیل .

روشن میشود؟

بله خیر

به‌سیستم‌های‌دیگر استارت‌میزند؟
رجوع کنید .

بله خیر

استارت‌تق‌صدا
میکند؟

بله خیر

سرباطری‌رابا بله نورچراغ خیر سرباطری‌شل
استارت‌یکسره زیاد است؟ نیست؟
کنید. میشود؟

بله خیر بله خیر

سوئیچ‌درست اتصال‌بدنه سرباطری‌را سیمها
است؟ درست‌است؟ محکم‌کنید. درست‌است؟

بله خیر بله خیر

خیر بله

اتومات سوئیچ‌را سیمهارا
را تعویض تعمیر کنید. تعمیرکنید. کنید .

اتصال‌بدنه استارت‌باید
رادرست‌کنید. بازشود .

شکل9/ضمیمه- نمودار فضای حالتهای عیب یابی سیستم استارت اتومبیل

متن برنامه

```
SIMPLE PROGRAM TO × ; PROGRAM STATUS_SPACE_SEARCH (× SHOW HOW USE THE)
                          (× SEM_NET TOOLS FOR STATUS_SPASE_SEARCH ×)
                                                              USES
                                                             , CRT
                                                           , SEM_NET
                                                           , SN_EXPL
                                                          ; SN_SERCH
                                                              CONST
                                (× NAME OF KNOWLEDGE BASE ×) ; 'CAR_REPR' = KB_NAME
                                                              TYPE
                                          (× A SIMPLE FRAME ×) RECORD = FRAME_REC
  CONTAIN MATER QUESTION OR ×) ; [STRING[100 : IF THIS NODE IS A
                                        MATTER_TEXT ×) ; BOOLEAN : GOAL_NODE (× SER
                                                 ; VAR END ; FILE OF FRAME_REC : FF

 FILE OF FRAMES CONTAIN INFOR ×) (× FRAME OF MSG TO U GOAL NODE ×) ;
FRAME_REC : FRAME (× MATION OF A NODE ; THIS PROCEDURE PROCEDURE LOADING
                                                                    ×
                                          (× LOAD THE KNOLEDGE BASE (× CURRENT NODE)
                                    (× WITH SIMPLE KNOLEDGES ABOUT START SYSTEM ×)
                                                          (× OF A CAR ×)
  ; STRING : M_T ; NAMES : NODE_NAME ) PROCEDURE SET_NODE   ( THIS
                                                 PROCEDURE MAKE A FRAME OF A ×
                                                      (× BOOLEAN STATE : GOAL)
                                                       (× AND MAKE ITS NODE ×)
                                                                  VAR
                                        ADDRESS OF FRAME IN FILE ×) ; LONGINT : ADDRES
```

```
                                                                »×
                                                             BEGIN
                                                  ; M_T =: FRAME.MATTER_TEXT
                                                   ; GOAL =: FRAME.GOAL_NODE
                        TAKE NEW FRAME IN END OF ×) ; ( FF ) FILESIZE =: ADDRES
                                                                 (× FILE FF
                                                      ; ( ADDRES , FF ) SEEK
                                WRITE NEW FRAME IN FILE ×) ; ( FRAME , FF ) WRITE
          MAKE NODE ×) ; ( ADDRES , NODE_NAME ) DEFINE_NODE (×  RESS ; END (×
                                                  OPENNING SYSTEM BEGIN ×
                                                       (× WITH FRAME ADD)
                                          ; ( KB_NAME ) SET_SN_ES_SYSTEM_ON
                                            ; ('FRM.'+KB_NAME , FF ) ASSIGN
                                                (× MAKE FRAME FILE ×) {-I$}
                                                           ; ( FF ) REWRITE
                                                    (× DEFINE EDGE_KINDS ×)
                                             ; ( 'TRUE' ) DEFINE_EDGE_KIND
                                            ; ( 'FALSE' ) DEFINE_EDGE_KIND
                                               DEFINE EDGE EXPLAIN SENT ×)
                              , 'IS TRUE SO' , 'TRUE' ) SET_EXPL_EDGE (× ENCES
                                                  ; ( 'BECAUSE ITS TRUE THE'
                                   , 'IS NOT TRUE SO' , 'FALSE' ) SET_EXPL_EDGE
                                               ; ( 'BECAUSE ITS NOT TRUE THE'
                                               DEFINE ALL STATUS AND IT ×)
          CAN THE CAR' , 'SET_ON' ) SET_NODE (× CAN THE CAR S FRAMES AND NODES'
                                            , 'DO_START' ) SET_NODE ; ( FALSE
                                          , 'TICK_START' ) SET_NODE ; ( FALSE
         CAN THE STARTER TAKE A TICK' , '? TAKE START , '? CAN THE BATTERY TAKE
                                           ON' , 'LOOSE' ) SET_NODE ; ( FALSE
                                          CAN' , 'LIGHT' ) SET_NODE ; ( FALSE

         FALSE , '? THE GRATE LIGHT IS HIGH , '? HEAD IS LOOSE , '? SOUND  ; (
                                                                       TRUE
         ) SET_NODE , '. KEEP THE BATTERY HEAD' , 'KEEP' ) SET_NODE ; ( ,
                                             'WIRE_REP' ) SET_NODE ; ( FALSE

         , '.REPAIRE OR CHANGE THE WIRES' , '? WIRES IS TRUE' , 'WIRE' OF+ TAKE
        A CONNECTION FROM' , 'CONNECT' ) SET_NODE ; ( TRUE  ) SET_NODE ; ( FALSE
                                                    , '?CAN START. RTER

                  , '?TEST THE SWITCH.ITS TRUE' , 'SWITCH_TEST' BATTERY TO STA
         TRUE , '.REPAIR THE SWITCH' , 'SWITCH_REP' ) SET_NODE ; ( REPAIR OR
        CHANGE THE FALSE' , 'START_AUTOMAT' ) SET_NODE ; ( , 'OPEN_START' )
                                                       SET_NODE ; ( TRUE
                                                                ; ( TRUE
                        , '.OPEN_AND REPAIR THE START' , 'AUTOMAT OF START  TRUE
         ; ( , '.CHECK OTHER SYSTEMS' , 'OTHER_SYSTEM' ) CHECK THE BODY
        CONNECT.IS IT SET_NODE' , 'BODY_CNCT' ) REPAIR THE SET_NODE' ,
                                          'BODY_CNCT_REP' ) SET_NODE ; ( FALSE
                                               DEFINE ALL EDGE AND × ; ( TRUE
                                             PATH , '.BODY_CONNECT , '?TRUE)

                  ; ( 0 , 'TRUE' , 'OTHER_SYSTEM' , 'SET_ON' ) DEFINE_EDGE (× S
                         ; ( 0 , 'FALSE' , 'DO_START' , 'SET_ON' ) DEFINE_EDGE
                     ; ( 0 , 'TRUE' , 'OTHER_SYSTEM' , 'DO_START' ) DEFINE_EDGE
                      ; ( 0 , 'FALSE' , 'TICK_START' , 'DO_START' ) DEFINE_EDGE
                          ; ( 0 , 'TRUE' , 'LIGHT' , 'TICK_START' ) DEFINE_EDGE
                        ; ( 0 , 'FALSE' , 'LOOSE' , 'TICK_START' ) DEFINE_EDGE
```

```
                        ; ( 0 , 'TRUE' , 'KEEP' , 'LOOSE' ) DEFINE_EDGE
                      ; ( 0 , 'FALSE' , 'WIRE' , 'LOOSE' ) DEFINE_EDGE
                    ; ( 0 , 'TRUE' , 'CONNECT' , 'LIGHT' ) DEFINE_EDGE
                     ; ( 0 , 'FALSE' , 'LOOSE' , 'LIGHT' ) DEFINE_EDGE
                       ; ( 0 , 'TRUE' , 'LIGHT' , 'WIRE' ) DEFINE_EDGE
                   ; ( 0 , 'FALSE' , 'WIRE_REP' , 'WIRE' ) DEFINE_EDGE
               ; ( 0 , 'TRUE' , 'SWITCH_TEST' , 'CONNECT' ) DEFINE_EDGE
                ; ( 0 , 'FALSE' , 'BODY_CNCT' , 'CONNECT' ) DEFINE_EDGE
           ; ( 0 , 'TRUE' , 'SWITCH_REP' , 'SWITCH_TEST' ) DEFINE_EDGE
 ( 0 , 'FALSE' , 'START_AUTOMAT' , 'SWITCH_TEST' ) DEFINE_EDGE ; ( 0 ,
                    'TRUE' , 'OPEN_START' , 'BODY_CNCT' ) DEFINE_EDGE
                                                                    ×
; ( 0 , 'FALSE' , 'BODY_CNCT_REP' , 'BODY_CNCT' ) DEFINE_EDGE ; (×
                                                    CLOSING SYSTEM)
                                                ; SET_SN_ES_SYSTEM_OFF
                                                        ; ( FF ) CLOSE
                                                             ; END
                                                ; PROCEDURE FIRST_LOADING
   THIS PROCEDURE MAKE A KNOLEDGE BASE AND THEN LOAD IT × WITH DEFINE)
                                                          (× KNOLEDGE
                                                               BEGIN
                                                ; ( KB_NAME ) DEFINE_SN_ES
                                                          ; LOADING
                                                             ; END
                                    ; BYTE : ( NAMES : NODE ) FUNCTION TRAVEL_STRATEGY
                                              THIS PROCEDURE IS MAIN PART OF TRAVE ×)
                                    IT RETURNED 4 DEFF - TROUGH THE STATES × (× LLING)
                                                  1 FOR TRUE EDGE : VALUE × (× ERENT)
                                                                              »×
                                                       2 FOR FALSE EDGE ×)
                                                                              »×
                                                  3 FOR RECIVING THE GOAL ×)
                                                                              »×
                                                    4 FOR ERROR IN ADDRES ×)
                                                                           (×
                                                                        VAR
                                                          ; BYTE : RESULT
                                                              ; CHAR : CH
                                                     ; LONGINT : ADDRES
                                                                    BEGIN
 GETTING ADDRESS OF ×) ; ( NODE ) USER_ADDR =: ADDRES  1 THEN IN- <>
                                        IF ADDRES (× EXIST FRAME FILE ×) BEGIN
                                          {-I$} LOAD ×) ; ( ADDRES , FF ) SEEK
                                                  ; ( FRAME , FF ) IF READ
NOT FRAME.GOAL_NODE (× FRAME (× SO ASK F FRAME A REAL ADDRESS - NOT
                                                GOAL NODE ×) OR TRUE OR FALSE BEGIN
( '(YES OR NO )' , FRAME.MATTER_TEXT ) WRITELN (× THEN EDGE  REPEAT  ;
                                ( READKEY ) UPCASE =: CH ) OR ( 'Y' = CH ) UNTIL
                                    THEN 'Y' = IF CH (× TRUE EDGE ×) 1 =: RESULT
                                         (× FALSE EDGE ×) ; 2 =: ELSE END RESULT
                                                                    ELSE END
×) ; 4 =: RESULT (× GOAL NODE ×) ; 3 =: ELSE RESULT ; ( 'N' = CH ; (×
                                                  RESULT =: TRAVEL_STRATEGY
                                                  ; PROCEDURE PASSENGER END
VAR ; NAMES : START_STATE ) ; ERROR IN ADDRES OF FRAME  × THIS PROCEDURE
                                                               MAKE TRAVEL)
  BY USING THE × (× BETWEEN STATES ; ( BOOLEAN : TRAVEL STRATEG ERROR)
                                                                  VAR (× Y
```

```
                                                    ×) ,CURRENT_NODE
                                              NAME OF NEXT NODE ×) , NODE_NAME
                                                        KIND ×) , EDGE_KIND
                                                                     EDGE
 : (× OF THIS EDGE (× OF THIS EDGE (× WEIGHT CURRENT NODE NAME ×) ;
                                                        INTEGER : WEIGHT
                                                        FOUND ×) , FOUND

 (× THE ASKED EDGE (× OF EDGE (× ASKED EDGE KIND ×) ; NAMES ×) ; BOOLEAN
                                                                : DIRECT

        (× DIRECT OF THIS EDGE (× FOUND THE GOAL ×) , BEGIN GOAL_FOUND
 (× ACTION BY TRAVEL STRATEGY ×) ; BYTE : IN FIRST BEGIN AT START ACTION
×) ; START_STATE =: RESET AL PASS NODE FLAG TO CURRENT_NODE ×) ;
                                                        RESET_PASS_NODES
                                    ; FALSE =: ERROR (× TION OF LOOP SEARCH
                                              ; FALSE =: GOAL_FOUND =: ACTION
 ( CURRENT_NODE ) PREVEN REPEAT TRAVEL_STRATEGY (× STATE  CASE ACTION
                                                    OF (× EDGE RATEGY : 1
                                                ; 'FALSE' =: EDGE : 2 : 3
                                                                     : 4
 ; TRUE =: GOAL_FOUND ; 'TRUE' =: TAKE THE TRAVEL ST ×) ; ; NOT END )
                                                  AND ( NOT GOAL_FOUND ) IF
                                                    SERACH AND ×) BEGIN
                                                     ) IF FIND_NODE (×
 THEN ( THEN TRAVEL CURRENT_NODE ( ERROR ; TRUE =: ERROR  ALLREADY ×)
                                                  ;TRUE =: ERROR (× ODE
                                               MAKE A LOOP - THIS NODE
                                                     ) FIND_FIRST_EDGE
 , EDGE_KIND , NODE_NAME (× FIND THE CURRENT N PASS FROM ×)  ; FALSE =:
                                                FOUND = SN_KB_ERROR ) WHILE
 ( NOT ERROR ) AND ( NOT FOUND ) AND ( 0 ; ( DIRECT , LOOP FOR FOUND ASK
                                    WEIGHT ×) EDGE_KIND BEGIN ) IF (× ED EDGE
                                              FOUND ASKED EDGE ×) BEGIN
                                                     ( ) SIMPLE_EXPLAIN
                                    × AND DIRECT THEN ( EDGE = DO  EXPLAIN ×
                                (× MAKING ; ( DIRECT ,CURRENT_NODE,NODE_NAME,EDGE)
                                 MAKE A STATE TRAVE ×) ; NODE_NAME =: CURRENT_NODE
                          »; TRUE =: FOUND × (CHANGE NODE TO NEXT NODE) L STEP
                                                                ; END
  , WEIGHT , EDGE_KIND , NODE_NAME ) FIND_NEXT_EDGE  ; NOT NOT FOUND
                                               THEN IF END ×) ; TRUE =: ERROR
                                             ; END (× UNTIL GOAL_FOUND OR GE
                                                      ; PROCEDURE USING END
                                               THIS PROCEDURE MAKE A ×
          (× SEARCH IN STATE SPACE ; ; EXIST ASKED ED ERROR ;( DIRECT)
                                                                   VAR
                                                  ; NAMES : START_STATE
                                                      ; BOOLEAN : ERROR
                                                                   BEGIN
                                                                ; CLRSCR
                                            ; ( KB_NAME ) SET_SN_ES_SYSTEM_ON
                                             ; ( 'FRM.'+KB_NAME , FF ) ASSIGN
                                                                   {-I$}
                                                          ; ( FF ) RESET
                                                  0 THEN <> IF IORESULT
                                                                ; EXIT
                                             (× DEFINE START_STATE ×) REPEAT
```

```
                                                    ; ( '? WHAT IS PROBLEM' ) WRITELN
                                                              ; ( START_STATE ) READLN
                                                          ; ( START_STATE ) FIND_NODE
                                                             0 THEN <> IF SN_KB_ERROR
                                              ; ( '.SORRY.NOT EXIST THIS PROBLEM IN KB' ) WRITELN
                                                             ; 0 = UNTIL SN_KB_ERROR
 CALL TRAVEL ×) ; ( ERROR , START_STATE ) PASSENGER  IF ERROR THEN
                                                                    TATUS (× SPACE
                                                                             ELSE
) WRITELN ( 'ERROR IN KB' ) PASENGER ROTIN IN S WRITELN ; ( FF ) COPY
                                                                CLOSE ×) ; COPY_EXPLAIN
                                                                                (× T
EXPLAINS TO FILE EXPLAIN.TX ; ( FRAME.MATTER_TEXT  ; END× (× MAIN)
                                                                              BEGIN
ONLY IN FIRST ×) ;FIRST_LOADING ; SET_SN_ES_SYSTEM_OFF (× IN ALL RUNNING
                                                        > (×پایان‌صفحه ;USING  .END <
```

RUNNING ×) ضمیمه 2 - نمونه محاورات‌انجام شده توسط برنامه سمپاد 1 .

تذکر :

نوشته های سمت راست از طرف سیستم و نوشته‌های سمت چپ از طرف کاربر میباشد .

سلام - به‌سیستم سمپاد 1 خوش‌آمدید .

روی چه‌پایگاهی میخواهید کار کنید ؟

داده ها

چنین پایگاهی موجود نیست.آنرا بوجود بیاورم؟

بله

پایگاه ایجاد شد .

پایگاه بارگیری شد .

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :

کارمند موجودیت‌جدید ایجادکنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :

علی

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :

محمد موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :

مهدی

نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :

هست نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :

فرزند نوع ارتباط جدید ایجاد کنید

ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :

مرد وضعیت‌ارتباط را تعیین کنید وضعیت‌چه‌ارتباطی‌تعیین‌شود؟

هست آیاارتباط توارث‌مستقیم‌دارد؟

بله

آیاارتباط توارث‌معکوس‌دارد؟

بله

آیاارتباط یکتائی مستقیم‌دارد؟
خیر آیاارتباط یکتائی معکوس‌دارد؟
خیر وضعیت‌ارتباط را چاپ‌کنید وضعیت‌چه‌ارتباطی‌چاپ‌شود؟
هست توارث‌معکوس_ توارث‌مستقیم_ وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
هست توضیح ارتباط درحالت مستقیم چیست؟
هست‌یک توضیح ارتباط درحالت معکوس چیست؟
وجود دارد به نام
وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
فرزند توضیح ارتباط درحالت مستقیم چیست؟
فرزند کسی است‌به نام
توضیح ارتباط درحالت معکوس چیست؟
فرزندی دارد به نام
وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
مرد توضیح ارتباط درحالت مستقیم چیست؟
مردی است‌با نام
توضیح ارتباط درحالت معکوس چیست؟
مردی است‌با نام
وضعیت‌توضیح ارتباط را چاپ‌کنید توضیح چه ارتباطی چاپ شود؟
فرزند معکوس:فرزندی دارد به نام مستقیم:فرزند کسی است‌به نام
ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
علی
به چه موجودیتی ارتباط برقرار شود ؟
کارمند چه نوع ارتباطی‌برقرار شود؟
هست باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (

ESC ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
محمد به چه موجودیتی ارتباط برقرار شود ؟
مهدی
چه نوع ارتباطی‌برقرار شود؟
فرزند باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (
ESC ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
علی
به چه موجودیتی ارتباط برقرار شود ؟
محمد چه نوع ارتباطی‌برقرار شود؟
فرزند باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (
ESC ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
کارمند به چه موجودیتی ارتباط برقرار شود ؟
ESC
ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟

علی

به چه موجودیتی ارتباط برقرار شود ؟

کارمند چه نوع ارتباطی‌برقرار شود؟

هست باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (

ESC وجود داشته .

ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟

مهدی

به چه موجودیتی ارتباط برقرار شود ؟

مهدی

چه نوع ارتباطی‌برقرار شود؟

مرد باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (

ESC نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :

پدربزرگ ارتباط مرکب‌ایجاد کنید کدام ارتباط به‌ارتباط مرکب تبدیل شود ؟

پدربزرگ بترتیب ارتباط توالی را ازآخر به‌اول‌بنویسیدودرانتهای‌توالی‌باESCخارج‌شوید. ارتباط ساده توالی را بنویسید :

مرد آیا ارتباط درحالت مستقیم است ؟

بله

ارتباط ساده توالی را بنویسید :

فرزند آیا ارتباط درحالت مستقیم است ؟

خیر ارتباط ساده توالی را بنویسید :

فرزند آیا ارتباط درحالت مستقیم است ؟

خیر ارتباط ساده توالی را بنویسید :

ESC خداحافظ به همین‌زودی خسته شدید ؟ یا کارتان تمام‌شد ؟

بله

به امید دیدار مجدد .

--سلام - به‌سیستم سمپاد _1 خوش‌آمدید .

روی چه‌پایگاهی میخواهید کار کنید ؟

داده ها

پایگاه بارگیری شد .

وضعیت‌توضیح ارتباط را چاپ‌کنید توضیح چه ارتباطی چاپ شود؟

پدربزرگ معکوس:پدربزرگی دارد به نام مستقیم:پدر بزرگ‌کسی است‌به نام

جستجو کنید

از چه‌موجودیتی جستجو کنم؟

کارمند چه‌نوع ارتباطی را بیابم؟

پدربزرگ آیا ارتباط درحالت مستقیم جستجو شود ؟

خیر

آیا توارث درحالت مستقیم تعیین شود ؟

خیر باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (

ESC مهدی

توضیح دهید به دلیل اینکه کارمند وجود دارد به نام علی پس

یک پدربزرگ مردی استبا نام فرزندی دارد به نام فرزندی دارد به نام
آناست
علی فرزند کسی استبه نام محمد محمد فرزند کسی استبه نام مهدی مهدی مردی استبا نام مهدی علی پدربزرگی دارد به نام مهدی خداحافظ
به همین‌زودی خسته شدید ؟ یا کارتان تمام‌شد ؟
بله
به امید دیدار مجدد .

< پایان‌صفحه>

ضمیمه 3 - روالها و عملیات عمومی ابزار .

کاربر برنامه نویس این ابزار روالهای زیر را برای ایجاد و استفاده سیستم خبره در اختیار دارد .

1- روالهای ایجاد و استفاده از پایگاه دانش .

تعریف نوع ارتباط (لبه) جدید .
DEFINE_EDGE_KIND
تعریف ارتباط (لبه) جدید .
DEFINE_EDGE
تعریف موجودیت (گره) جدید .
DEFINE_NODE
حذف ارتباط .
DELETE_EDGE
حذف موجودیت .
DELETE_NODE
یافت یک موجودیت با نام معین و تعیین‌آن به عنوان موجودیت جاری .
FIND_NODE
یافت اولین لبه متصل به موجودیت جاری .
FIND_FIRST_EDGE
یافت لبه بعدی متصل به موجودیت جاری .
FIND_NEXT_EDGE
تعیین وضعیت نوع لبه (توالی - جهت - توارث) .
SET_EDGE_KIND_STATUS
تعریف و اضافه سازی مرحله توالی نوع لبه .
ADD_EDGE_SEQUENCE
دریافت وضعیت لبه .
GET_EDGE_KIND_STATUS

2- روالهای استفاده از مکانیزم‌های جستجو .

روال تعریف یک سیستم خبره جدید .
DEFINE_SN_ES
روال روشن کردن سیستم خبره (در ابتدای‌کار)
SET_ES_SYSTEM_ON
روال خاموش کردن سیستم خبره

SET_ES_SYSTEM_OFF

روال جستجوی ساده

SIMPLE_SEARCH

روال جستجوی توارثی

INHERITNESS_SEARCH

روال جستجوی توالی

SEQUENCE_SEARCH

روال جستجوی عمومی (ترکیب توارث‌و توالی)

GENERAL_SEARCH

روال تعیین دستگاه خروجی (مانیتور-چاپگر- فایل)

SET_OUTPUT_DEVICE

-3 روالهای استفاده از سیستم توضیح .

روال تعیین عبارات ربط توضیح

SET_EXPL_EDGE_PARAMETER

روال دریافت عبارات ربط توضیح

GET_EXPL_EDGE_PARAMETER

روال کپی و اعلان توضیح‌بر خروجی

COPY_EXPLAIN

روال پاک کردن توضیحات قبلی

CLEAR_EXPLAIN

-4 مثالی‌از یک برنامه نمونه-ارتباطات کارمندان وخانواده‌آنها .

PROGRAM TEST_SN_KB ; (× این برنامه چگونگی استفاده از روالهای ابزار (× SEM_NET ×) را نشان میدهد (× .

```
USES
, SN_SERCH
, SN_EXPL
; SEM_NET
; PROCEDURE INIT_KB
```

(× این زیر برنامه در ابتدا پایگاه دانش سیستم خبره را تعریف میکند (×

```
BEGIN
' ) ; رابطه‌هاDEFINE_SN_ES ( '
; END
; PROCEDURE LOADING
```

(× این زیر برنامه پایگاه دانش را بارگیری میکند (× .

```
BEGIN
```

SET_SN_ES_SYSTEM_ON('رابطه‌ها'); (× روشن کردن سیستم خبره ×)

DEFINE_EDGE_KIND('هست‌یک'); (× تعریف‌نوع ارتباطات ×)

```
' ) ; فرزندDEFINE_EDGE_KIND ( '
' ) ; مردDEFINE_EDGE_KIND ( '
' ) ; پدربزرگDEFINE_EDGE_KIND ( '
```

```
(× تعریف توضیح ارتباطات
' ) ; هست‌یکی‌از' ,'هست‌یک' ,'هست‌یکSET_EXPL_EDGE ( '
' ) ; فرزندی‌داردبه‌نام' ,'فرزندکسی‌است‌به‌نام' ,'فرزندSET_EXPL_EDGE ( '
' ) ; مرداست' ,'مرداست' ,'مردSET_EXPL_EDGE ( '
' , پدربزرگ‌کسی‌است‌به‌نام' ,'پدربزرگSET_EXPL_EDGE ( '
' ) ; پدربزرگی دارد به‌نام'
(× تعیین وضعیت لبه
' , TRUE , FALSE , FALSE , FALSE ) ; هست‌یک ×) SET_EDGE_KIND_STATUSتعریف
گره‌ها ×);( 0 ,'علی' , 0 ) ; DEFINE_NODE ( 'رضاDEFINE_NODE (
' , 0 ) ; تقی' , 0 ) ; DEFINE_NODE ( 'کریم' , 0 ) DEFINE_NODE ( '
( ' ' ; مهدیDEFINE_NODE ( '
' , 0 ) ; نقیDEFINE_NODE ( '
' , 0 ) ; داودDEFINE_NODE ( '
' , 0 ) ; کارمندDEFINE_NODE ( '
(× تعریف‌توالی پدربزرگ
' , TRUE ) ; مرد' ,'پدربزرگADD_EDGE_SEQUENCE ( '
' , TRUE ) ; فرزند' ,'پدربزرگADD_EDGE_SEQUENCE ( '
' , TRUE ) ; فرزند' ,'پدربزرگADD_EDGE_SEQUENCE ( '
(× تعریف ارتباطات
' , 0 ) ; فرزند' ,'کریم' ,'تقیDEFINE_EDGE ( '
' , 0 ) ; فرزند' ,'تقی' ,'نقیDEFINE_EDGE ( '
' , 0 ) ; فرزند' ,'رضا' ,'علیDEFINE_EDGE ( '
' , 0 ) ; فرزند' ,'مهدی' ,'رضاDEFINE_EDGE ( '
' , 0 ) ; فرزند' ,'مهدی' ,'داودDEFINE_EDGE ( '
' , 0 ) ; مرد' ,'مهدی' ,'مهدیDEFINE_EDGE ( '
' , 0 ) ; مرد' ,'کریم' ,'کریمDEFINE_EDGE ( '
' , 0 ) ; هست‌یک' ,'کارمند' ,'علیDEFINE_EDGE ( '
' , 0 ) ; هست‌یک' ,'کارمند' ,'نقیDEFINE_EDGE ( '
; SET_SN_ES_SYSTEM_OFF
; END
; PROCEDURE SEARCH
(× زیر برنامه انجام چند جستجو
VAR
; NAMES : E , N
; INTEGER : W
; BOOLEAN : D
BEGIN
' ) ; رابطه‌هاSET_SN_ES_SYSTEM_ON ( '
; ( '--------------------------------' ) WRITELN
' ) ; فرزندان مهدیWRITELN ( ' :
' , FALSE , 0 ) ; فرزند' ,'مهدیSIMPLE_SEARCH ( '
; ( '--------------------------------' ) WRITELN
' ) ; لیست پدر تمام‌کارمندانWRITELN ( ' :
```

```
' ) INHERITANCE_SEARCHکارمند','فرزند' ; ( 0 , TRUE , FALSE , '
; ( '------------------------------' ) WRITELN
' ) ; لیست تمام کسانی که کریم پدربزرگ آنها است' ) WRITELN (
' , TRUE ) ; پدربزرگ','کریم' ) SEQUENCE_SEARCH (
; ( '------------------------------' ) WRITELN
' ) ; لیست پدربزرگ تمام کارمندان' ) WRITELN ( ' :
' ) GENERAL_SEARCHکارمندان','پدربزرگ' ; ( 0 , FALSE , FALSE , '
(× توضیح COPY_EXPLAIN ; (×
; SET_SN_ES_SYSTEM_OFF
; END
(× برنامه اصلی (×
BEGIN
(× تعریف پایگاه - فقط بار اول اجرا شود INIT_KB ; (×
(× بارگیری پایگاه - فقط یک‌بار اجرا شود LOADING ; (×
(× جستجو SEARCH ; (×
.END
```


3- بررسی ابزار ایجاد سیستم‌خبره مبتنی بر شبکه معنائی .

در این بخش به بررسی ابزار ایجاد سیستم خبره مبتنی بر شبکه معنائی میپردازیم .

3-1 سیستمهائی که بوسیله این ابزار پیاده میشوند .

هر سیستم خبره که دانش مربوط به‌آن قابل نمایش توسط شبکه معنائی باشد قابل پیاده سازی توسط این ابزار است . با توجه به این موضوع تعداد زیادی از مسائل و سیستم هامیتوانند تحت پوشش این ابزار قرار گیرند زیرا بسیاری از مسائل کاربردی ماهیت شبکه‌ای و گراف شکل دارند . ازآن جمله میتوان به‌موارد زیر اشاره نمود :

قطعات مورد نیاز برای ساخت یک محصول در یک کارخانه تولیدی در یک سیستم انبارداری .

تمام ادارات ، قسمتها ، کارمندان ، عمل‌ها در یک سیستم
اداری و یا صنعتی .

مسیرها در مسئله ترابری .

پیمایش گراف تشخیص (نمونه در بخش 2-7-3) .

کلیه ساختارهای درختی .

پیمایش فضای حالتهای گراف شکل .

سیستم‌های مترجم زبانهای انسانی .

بطور کلی با توجه به اینکه میتوان اغلب روشهای نمایش
دانش را به گونه ای به شبکه معنائی تبدیل نمود (از جمله

PRODUCTION SYSTEMS و FRAME و (SCRIPT این ابزار را میتوان
پوشا بر اغلب مسائل سیستم‌های خبره دانست .

2-3- پایگاه دانش .

1-2-3- شیوه نمایش دانش .

با توجه به مطالب بیان شده در بندهای قبلی علاوه بر
شیوه نمایش مبنای شبکه معنائی دو شیوه نمایش دانش FRAME و

SCRIPT را به صورت مستقیم و شیوه نمایش دانش PRODUCTION -
SYSTEMS را به صورت ضمنی در این پایگاه نمایش داد .

2-2-3- حجم دانش .

حجم دانش ذخیره شده در سیستم SEM_NET از لحاظ تعداد
موجودیت محدود به 1024 موجودیت، ولی سیستم SEM_NETL محدودیتی

از نظر تعداد موجودیت ندارد . هیچیک از سیستم‌های ذکر شده از نظر تعداد ارتباطات محدودیتی ندارند اما ایجاد تعداد زیادی ارتباط (بیش از 100) بر روی یک موجودیت باعث افت کارائی و در اینگونه موارد استفاده از روشهای مناسب در تعریف موجودیت ها و نوع ارتباطات میتواند از پیش‌آمدن این موضوع جلوگیری نماید .

حجم تعداد نوع ارتباط محدود به عدد 225 است .

%
37

3-2-3- پویائی دانش .

دانش موجود در پایگاه دانش از نظرات زیر پویا است :

1- پویائی از نظر تعداد موجودیت‌ها(ایجاد-حذف) .

2- پویائی از نظر تعداد نوع ارتباطات(ایجاد) .

3- پویائی از نظر تعداد ارتباطات(ایجاد-حذف) .

4- پویائی از نظر محتوای موجودیت‌ها (شکل و اندازه رکورد و ساختمان موجودیت مثلا FRAME یا . (SCRIPT

5- پویائی از نظر محتوای ارتباطات(وزن ارتباطات) .

6- پویائی از نظر خصوصیات نوع ارتباط (توارثی - جهت) .

7- پویائی از نظر توالی منطقی زنجیره توالی (ایجاد توالی) .

8- پویائی از نظر ماهیت و نوع ارتباطات بین موجودیت‌ها .

3-3- مکانیزم‌های جستجو .

1-3-3- انعطاف‌پذیری مکانیزم‌ها .

انعطاف پذیری مکانیزم‌ها از دو جنبه قابل بررسی است .

3-3-1-1- انعطاف پذیری مکانیزم‌های جستجو .

پارامترهای جستجو انعطاف‌پذیری جستجو را افزایش
%
38

میدهند . پارامترهائی که در جستجو نقش دارند عبارتند از :

موجودیت‌آغازین .

نوع ارتباطات .

وزن ارتباطات .

جهت ارتباطات .

جهت توارث .

جهت توالی .

3-3-1-2- قابلیت طراحی واحد استنتاج .

واحد استنتاج با تعداد دستورات بسیار کم‌و با ساختمان نسبتا ساده توسط استفاده کننده قابل طراحی است . نمونه واحد استنتاج در واحد استنتاج برنامه عیب یاب اتومبیل قابل مشاهده است .

3-3-2- تعداد مکانیزم‌ها

مکانیزم‌های جستجو زیر قابل استفاده اند .

ساده

توارثی

توالی

3-3-3- ایجاد مکانیزم‌های جدید توسط کاربر .

با در اختیار قرار گرفتن زیر روالهای یافت گره جاری ، یافت اولین ارتباط و یافت ارتباط بعدی که در روالهای مربوط به پایگاه‌دانش موجود است ، استفاده کننده میتواند روالهای جستجوی مورد نیاز خود را خود طراحی نماید .

4-3- سیستم توضیح .

1-4-3- پویائی سیستم توضیح .

وجود قابلیت تعریف و تغییر عبارات توصیفی موجب ایجاد پویائی سیستم توضیح نسبت به تغییر عبارات (در اصلاح عبارات) و تغییر نوع و ماهیت ارتباطات (در اصلاح یا تعریف نوع ارتباطات) است .

2-4-3- مکانیزم ساخت جملات .

مکانیزم ساخت جملات با توجه به‌آنکه تنها از ترکیب نام موجودیت اول به علاوه عبارت ربط و موجودیت دوم تشکیل میگردد مکانیزم چندان قوی به‌نظر نمیرسد . اما سادگی مکانیزم به‌آن منجر میشود که تعریف کننده پایگاه دانش به سادگی بتواند چگونگی جمله و عبارات ربطی مربوط به

ارتباط مورد نظر را برای سیستم تنها با یک عبارت‌در حالت مستقیم و یک عبارت در حالت معکوس تعریف نماید .

هر چند جملات ساخته شده در این عبارت چندان دلچسب
نیستند اما معنای مورد نظر را میرسانند .

3-4-3- جمله سازی فارسی در سیستم توضیح .

سیستم این امکان را میدهد تا جملات در دو وضعیت فارسی و
یا لاتین شکل بگیرد و این کار با تعریف یک متغییر در
ابتدای کار به یکی از دو صورت زیر امکان پذیر است :

```
; FARSI =: EXPLAIN_TEXT_FARSI_LATIN
```

و یا :

```
; LATIN =: EXPLAIN_TEXT_FARSI_LATIN
```

این متغییر ماهیت جمله بندی را و نیز کلماتی که در
جمله بندی توسط سیستم مورد استفاده قرار میگیرد (نظیر
چون ، پس ، BECAUSE، (SOرا تغییر میدهد و بنابراین از
سیستم توضیح هم در محیط فارسی و عربی و هم در محیط لاتین
میتوان استفاده نمود .

3-5- بررسیابزار از نظر پارامترهای کیفیت نرمافزار .

انعطاف پذیریFLEXIBLITY

انعطاف پذیری از جنبههای زیر قابل بررسی است(که
%

توضیحاتآن در بندهای گذشته ارائه گردید : (

انعطافپذیری پایگاه دانش .

انعطافپذیری مکانیزمهای استنتاج .

انعطافپذیری سیستم توضیح .

قابلیتآزمونTESTABILITY

وجود توابعی که مقادیر و وضعیت موجود پایگاه دانش را

باز میگردانند در سیستم (مثلابازگرداندن وضعیت نوع لبه)
و نیزوجود مکانیزم یافت خطا درسیستم با متغییر SN_ES_ERROR
و تعیین شدن این متغییر در تمام عملیات داخلی ابزار
و تابعی که متن خطای مربوطه را به فارسی یا لاتین
بازمیگرداند به‌آزمون نرم‌افزار کمک میکند .

قابلیت حملPORTABILITY

با وجود فایل OBJ ابزار این ابزار قابل استفاده بر
روی تمام نسخه‌های (VERSIONS) کامپایلر BORLAND TURBO-
PASCAL میباشد .

قابلیت استفاده مجددREUSABILITY

اصولا ماهیت یک ابزار مشخص کننده این قابلیت برای
استفاده در سیستم‌های نرم‌افزاری مختلف از یک نرم‌افزار
مشخص است .

قابلیت یادگیریLEARNABILITY

محدود بودن تعداد دستورات و نیز راهنمای استفاده و
مستندات استفاده سیستم و همچنین ساده بودن طریقه
استفاده و وجود چند برنامه نمونه ضریب یادگیری کار با
ابزار را افزایش میدهد .

قابلیت استفادهUSABILITY

قابل استفاده بودن سیستم در اکثر موارد مطرح شده
بنابر مندرجات بند 1-3 موید این خصوصیت میباشد .

کارائیEFFICIENCY

باآزمایشاتی که بر میزان سرعت و حجم حافظه مصرفی
سیستم انجام گردیده سرعت عملیات از سیستمهای سنتی
موجود بسیار سریعتر و حجم حافظه مصرفی اندکی افزونگی
دارد که با توجه به وضعیت عمومی سیستم قابل چشم پوشی
است .(محاسبه دقیق کارائی یک ابزار بنا بر حجم بالای
عملیات و بنا به ماهیت این مقاله در این مقال
نمیگنجد . (

قابلیت نگهداری سیستمMAINTAINABILITY .

بنا به سادگی روالهای مورد استفاده و مشخص بودن
وضعیت و ساختمان عمومی عملیات داخلی در مستندات
سیستم امکان نگهداری سیستم و توسعه سیستم بر روی ابزار
وجود دارد .

قابلیت اتصال به نرم افزار های دیگرINTERPERABILITY .

قابلیت استفاده این ابزار در یک زبان برنامه نویسی
این امکان را فراهم میاورد تا این سیستم بتواند با هر
سیستم دیگری که توسط این زبان ایجاد شده یا از طریق
کامپایلر امکان ایجاد ارتباط باآن وجود دارد ارتباط
ایجاد نماید و به سیستمهای دیگر متصل شود .

%

43

-3-6 کاربرد ابزار در پیمایش فضای حالتهادر سیستم‌های هوشمند .

-3-6-1 چگونگی انجام پیمایش فضای حالتها .

در صورتیکه در فضای حالتها هر حالت بعنوان یک موجودیت
(گره) و امکان حرکت از یک‌حالت به حالت دیگر با یک ارتباط
(لبه) تعریف گردد به سادگی میتوان با طراحی یک واحد
استنتاج مناسب از برنامه برای پیمایش این فضا استفاده نمود .
برنامه ذکر شده در قسمت بعد این موضوع را نشان میدهد .

-3-6-2 نمونه‌ای از پیمایش فضای حالتها در یک سیستم عیب یاب سیستم استارت اتومبیل .

این برنامه از تعدادی حالت تشکیل میگردد که هر حالت
بیانگر موقعیت خاصی از وضعیت امکان پذیر در جستجوی پیرامون
عیب سیستم استارت اتومبیل است . برنامه از یک حالت
ابتدائی که کاربرآنرا اعلام میکندآغاز میشود و تا رسیدن به

یک حالت جواب ادامه پیدا میکند . متن این برنامه و
توضیحات مربوط به‌آن در ضمیمه 1 وجود دارد .

%

44

4- بررسی یک سیستم پیاده سازی شده توسط ابزار سیستم‌سمپاد 1 .

4-1 هدف از ایجاد سیستم سمپاد 1 .

سیستم سمپاد 1 (سیستم محاوره‌ای پایگاه دانش) صرفا به عنوان وسیله
کمک‌آموزشی درآشنائی و ملموس شدن ماهیت سیستم‌های خبره و هوشمند
برای برنامه نویسان سنتی که میخواهند با روشهای هوش مصنوعی‌آشنا
گردند ایجاد گردید . این امر از دو طریق انجام میشود :

- 1 سیستم سمپاد 1 کلیه عملیات ایجاد پایگاه دانش و استفاده از
روالهای عملیات سیستم خبره را با محاوراتی با کاربر انجام
میدهد . این محاورات به گونه‌ای انجام میشود که کاربر با
ماهیت عملیات انجام شده توسط سیستم‌آشنا میگردد .

- 2 متن این برنامه (که تنها در حدود 200 خط برنامه است) در
اختیار علاقمندان قرار میگیرد تا هم با چگونگی استفاده از
ابزار ایجاد سیستم‌های خبره‌آشنا شوند و هم مکانیزم عملیات
یک برنامه هوش مصنوعی را مشاهده نمایند .

4-2 چگونگی عملکرد سیستم سمپاد 1 .

سیستم با انجام محاوراتی با کاربر ایجاد یک پایگاه دانش و
استفاده ازآنرا توسط کاربر انجام میدهد . سیستم تنها یک
برنامه واسط است که استفاده از روالهای ابزار را مستقیما به
کاربر محول میکند . در محاورات از چنین جملاتی استفاده میگردد :

موجودیت‌جدیدایجادکنید

%

نوع ارتباط جدید ایجاد کنید

وضعیت ارتباط را چاپ کنید

وضعیت ارتباط را تعیین کنید

وضعیت توضیح ارتباط را چاپ کنید

وضعیت توضیح ارتباط را تعیین کنید

ارتباط جدید ایجاد کنید

ارتباط را حذف کنید

موجودیت را حذف کنید

دستگاه خروجی را تعیین کنید

جستجو کنید

توضیح دهید

ارتباط مرکب ایجاد کنید

ننمونه‌ای از محاورات انجام شده با سیستم در ضمیمه 2 وجود دارد .

3-4- ساختمان عمومی سیستم سمپاد 1 و استفاده از ابزار ایجاد پایگاه دانش پویا .

سیستم از دو قسمت تشکیل میگردد :

1-3-4- بخش رابط کاربر

وظیفه این بخش انجام محاورات با کاربر است. روالهای این
بخش ماهیت سنتی دارند و از ساده‌ترین الگوریتمهای ممکن
برای سادگی یادگیری و جلوگیری از پیچیدگی تشکیل میشوند .

%

4-3-2 بخش کنترل و اعمال خواسته‌های کاربر

این بخش که در واقع بخش اصلی سیستم است وظیفه ترجمه خواسته های کاربر را به فراخوانی مناسب روالهای ابزار انجام میدهد . عملیات انجام شده به گونه‌ایست که دستوراتی را که کاربر صادر میکند با دستوراتی که برنامه به ابزار میدهد تناظر یک به یک دارد . از این رو ماهیت چگونگی انجام عملیات توسط ابزار برای کاربر سمپاد 1 روشن است .

%

5- آنچه گفته شد - نتیجه‌گیری .

5-1 چرا باید از ابزاردر ایجاد سیستم خبره استفاده نمود .

در ایجاد یک سیستم خبره دانش مربوطه توسط خبره‌انسانی بصورت قواعد ، تعاریف و .... بیان شده و مهندس دانش ، دانش بیان شده را به همراه داده‌های موجود در پایگاه داده‌ها (احتمالا) به شیوه نمایش مناسب در پایگاه دانش ذخیره نموده‌و مکانیزم‌های استنتاج را ایجاد مینماید .

در طراحی و ساخت روالهای ایجاد و استفاده پایگاه دانش و نیز طراحی و ساخت واحد استنتاج قسمت اعظم هزینه و زمان صرف طراحی و پیاده سازی روالهائی میگردد که ارتباط چندانی با موضوع خبرگی و هوشمندی نداشته و صرفا عملیات سیستمی و معمولی در تمام سیستمها است. ازجمله عملیات پردازش فایل و ذخیره سازی پایگاه دانش و جستجوی در اطلاعات موجود در پایگاه . انجام این عملیات نه‌تنها قسمت بسیار زیادی از هزینه و وقت طراحی

و پیاده‌سازی را به خود معطوف میکند بلکه ذهن طراحان را از تمرکز بر موضوع اصلی خبرگی باز داشته و متوجه سطوح زیرین

(نظیر عملیات در سطح فایل و رکورد) مینماید و از بهره‌وری

متمرکز فکر طراحان جلوگیری میکند . این موضوع باعث میشود تا لزوم وجود ابزار مناسب برای رهائی از سطوح زیرین و ایجاد سطوح انتزاعی مناسب احساس گردد .

یک ابزار ایجاد پایگاه دانش یک بسته نرم‌افزاری است که ساخت یک پایگاه دانش کاربردی راآسان میکند. این‌ابزارها به طراح کمک میکند که ساختمان پایگاه دانش ایجاد شده توسط ابزار

%

را به عنوان ساختمان مبنای کار خود قرار دهد و عملیات جستجوی در پایگاه دانش و استنتاجات بر مبنای این جستجو و نیز ارائه‌توضیح به کاربر را توسط روالهای این ابزار انجام دهد و تنها به چگونگی عملیات کلان برای ایجاد سیستم خبره بیاندیشد .

2-5- خصوصیات ابزار ایجاد سیستم‌های خبره مبتنی بر شبکه معنائی .

این‌ابزار به عنوان وسیله ایجاد سیستم‌های خبره مبتنی بر شیوه نمایش دانش شبکه معنائی با قابلیت پویائی در ایجاد و توسعه پایگاه دانش و مکانیزم‌های استنتاج بصورت محدود مورد استفاده قرار میگیرد . خصوصیات ابزار را میتوان در عبارات زیر خلاصه نمود :

قابلیت ایجاد و استفاده از پایگاه دانش به گونه‌ای که از شیوه نمایش دانش شبکه معنائی استفاده نماید و دانش موجود در پایگاه دانش قابلیت پویائی(از نظر حجم دانش ، ماهیت و نوع موجودیت‌ها ،ماهیت و نوع ارتباطات‌بین موجودیت‌ها ،محتوی موجودیت‌ها ، محتوی ارتباطات بین موجودیت‌ها) را داشته باشد .

وجود مکانیزم‌های جستجوی متفاوت (جستجوی ساده ، جستجوی توارثی بر مبنای توارث خصوصیات ، جستجوی توالی خاص ، جستجوی مرکب از دو حالت‌توالی‌خاص و توارث . (

مکانیزم‌های استنتاج با استفاده از مکانیزم‌های جستجو عمل مینمایند و قابلیت طراحی مکانیزم‌های استنتاج جدید با الگوربتمهای ساده امکان پذیر است .

قابلیت تعریف خصوصیات مربوط به جستجوها)
توارث، توالی ،ارتباطات‌عادی و جهت‌دارنمودن ارتباطات (
در پایگاه دانش .

قابلیت ارائه توضیح با دو نوع جمله بندی فارسی/لاتین
با قابل تعریف بودن عملگرهای توضیح با توجه به نوع
ارتباطات تعریف شده .

قابلیت اکتساب دانش و مکانیزم‌های جستجو در هنگام
اجرا و عملیات عمومی سیستم .

قابلیت ترکیب و استفاده از شبکه معنائی توام با روشهای
دیگرنمایش دانش) اختصاصا روش . (FRAME KN_PRESENTATION
قابلیت تعریف ، ایجاد ، استفاده‌و حذف یک پایگاه
دانش و سیستم خبره از داخل یک برنامه دیگر مثلا اگر
سیستم بنا به دلایلی به صورت موقت (TEMPORARY) ایجاد
شده باشد .

قابل استفاده بودن ابزار در زبان برنامه سازی PASCALو در
تمام سیستم‌های نرم‌افزاری که توسط کامپایلر زبان
برنامه‌سازی PASCAL ایجاد میگردند .

3-5- در چه شرایطی استفاده از این ابزار سودمند است .

این ابزار برای پیاده سازی هر سیستم خبره که دانش مربوط به
آن قابل نمایش توسط شبکه معنائی باشد و ماهیت شبکه‌ای و
گراف شکل یا درختی دارند مناسب است . با توجه به اینکه میتوان
اغلب روشهای نمایش دانش را به گونه ای به شبکه معنائی تبدیل

نمود این ابزار را میتوان برای اغلب مسائل سیستم‌های خبره
بکار گرفت .

4-5- نگاهی به ایجاد ابزارهای دیگر .

پیشرفت کاربرد سیستم‌های خبره در موارد حقیقی روزمره و نفوذ
روشهای هوشمند در ایجاد سیستمهای واقعی در سالهای اخیر بصورت
چشمگیری ملاحظه میشود . زبانهای برنامه سازی مناسب برای هوش
مصنوعی و سیستم‌های خبره نظیر PROLOG و LISP هنوز قابلیتها
و ابزارهای محیطی (ENVIRONOMENT TOOLS) مناسب برای ایجاد
سیستم‌های بزرگ و کاربردی را نیافته‌اند . در حالیکه زبانهای
برنامه‌سازی نظیر TURBO PASCAL و C++ با وجود امکانات جنبی و
محیطی مناسب بطور عادی برای پیاده سازی مکانیزم‌های هوشمندانه
مناسب نیستند . اینگونه مکانیزم‌ها برای پیاده سازی در این زبانها
تبدیل به عملیات بسیار پیچیده و پرحجمی در سطوح برنامه‌نویسی سنتی
میگردند . به همین دلیل اغلب در برنامه نویسی سیستم‌های بزرگ
برنامه‌نویس از خیر استفاده از روشهای هوش مصنوعی میگذرد . وجود
ابزارهائی نظیر ابزاری که در این مقاله برآن سخن رفت امکان
استفاده از روشهای هوش مصنوعی را در این زبانها ممکن و ساده
میسازد . استعداد نیاز به این ابزارها در زمینه‌های مختلف
هوش مصنوعی گسترده است و نگارندگان این مقاله حداقل 9 مورد
از نیاز به این ابزارها را در زمینه‌های مختلف احساس میکنند .
ولی ساخت این ابزارها میسر نیست به جز همت دانشگاهیان در
این امر . .....

مراجع .

ARTIFICAL INTELLIGENCE AND" . David W.Rolston : [ROLS88] . 1988.
McGRAW-HILL . " SYSTEMS DEVELOPMENT
INTEGRATING KNOWLEDGE-BASED" . Ruth Kerry : [RUTH90] EXPERT ELLIS
HORWOOD . " MANAGMENT SYSTMES
THE " . Erik Hollnagel : [ERIK89]

ELLIS . " RELIABILITY OF EXPERT SYSTEMS . AND DATABASE 1990 ARTIFICAL
" . Christopher F.Chabris : [CHAB89]
GALGOTIA .PASCAL
: [SCHA90]
" . Robert J.Shalkoff . TURBO 1989 & INTELLIGENCE . HORWOOD 1989 .
McGRAW-HILL 1990 : [HERB90]

ARTIFICIAL " . Herbert Schildt . " ARTIFICIAL INTELLIGENCE . McGRAW-HILL 1990 : [PRES87]
SOFTWARE " . Roger S.Pressman . " INTELIGENCE USING C . McGRAW_HILL 1987 : [BELL87]
SOFTWARE " . BELL-MORREY-PUGH . " ENGINEERING Prentice/Hall . 1987 " . Robin J.Wilson : [ROBN85]

% ENGINEERING " . INTRODOCTION TO GRAPH THEORY . " TECHNICAL 1985 & LONGMAN SCIENTIFIC

ضمیمه 1 - متن برنامه

عیب یاب سیستم استارت اتومبیل .

روشن میشود؟

بله خیر

به‌سیستم‌های‌دیگر استارت‌میزند؟
رجوع کنید .

بله خیر

استارت‌تق‌صدا
میکند؟

بله خیر

سرباطری‌رابا بله نورچراغ خیر سرباطری‌شل
استارت‌یکسره زیاد است؟ نیست؟
کنید. میشود؟

بله خیر بله خیر

سویچ‌درست اتصال‌بدنه سرباطری‌را سیمها
است؟ درست‌است؟ محکم‌کنید. درست‌است؟

بله خیر بله خیر

خیر بله

اتومات سویچ‌را سیمهارا

را تعویض تعمیر کنید. تعمیرکنید. کنید .

اتصال‌بدنه استارت‌باید

رادرست‌کنید. بازشود .

شکل9/ضمیمه- نمودار فضای حالتهای عیب یابی سیستم استارت اتومبیل

متن برنامه

```
                                        ; PROGRAM STATUS_SPACE_SEARCH
                                   (× SIMPLE PROGRAM TO SHOW HOW USE THE ×)
                                (× SEM_NET TOOLS FOR STATUS_SPASE_SEARCH ×)


                                                                  USES
                                                                , CRT
                                                              , SEM_NET
                                                              , SN_EXPL
                                                             ; SN_SERCH


                                                                  CONST
                                         (× NAME OF KNOWLEDGE BASE ×) ; 'CAR_REPR' = KB_NAME


                                                                  TYPE
                                                 (× A SIMPLE FRAME ×) RECORD = FRAME_REC
                                  CONTAIN MATER QUESTION OR ×) ; [STRING[100 : IF THIS NODE IS A
                                                       MATTER_TEXT ×) ; BOOLEAN : GOAL_NODE (× SER
                                                                     ; END   FILE OF VAR : FF
                                 FILE OF FRAMES ×) ; FRAME_REC (× MSG TO U GOAL NODE ×) ; FRAME_REC :
                                                                  FRAME (× MATION OF A NODE
                                                                                  PROCEDURE
                                                                                          ×
                                 THIS ; LOADING (× CONTAIN INFOR FRAME OF CURRENT NODE (× PROCEDURE LOAD
                                                                                  THE KNOLEDGE BASE)
                                                          (× WITH SIMPLE KNOLEDGES ABOUT START SYSTEM ×)
                                                                                  (× OF A CAR ×)


                                  ; STRING : M_T ; NAMES : NODE_NAME ) PROCEDURE SET_NODE   ( THIS
                                                                  PROCEDURE MAKE A FRAME OF A ×
                                                                       (× BOOLEAN STATE : GOAL)
                                                                        (× AND MAKE ITS NODE ×)

                                                                                      VAR
                                                          ADDRESS OF FRAME IN FILE ×) ; LONGINT : ADDRES
                                                                                      (×
```

```
                                                                BEGIN
                                                    ; M_T =: FRAME.MATTER_TEXT
                                                     ; GOAL =: FRAME.GOAL_NODE
                    TAKE NEW FRAME IN END OF ×) ; ( FF ) FILESIZE =: ADDRES
                                                                  (× FILE FF
                                                        ; ( ADDRES , FF ) SEEK
                          WRITE NEW FRAME IN FILE ×) ; ( FRAME , FF ) WRITE
    MAKE NODE ×) ; ( ADDRES , NODE_NAME ) DEFINE_NODE (×  RESS ; END (×
                                                          OPENNING BEGIN ×
                                                  (× WITH FRAME ADD SYSTEM)
                                        ; ( KB_NAME ) SET_SN_ES_SYSTEM_ON
                                          ; ('FRM.'+KB_NAME , FF ) ASSIGN
                                              (× MAKE FRAME FILE ×) {-I$}
                                                        ; ( FF ) REWRITE
                                                   (× DEFINE EDGE_KINDS ×)
                                                ; ( 'TRUE' ) DEFINE_EDGE_KIND
                                               ; ( 'FALSE' ) DEFINE_EDGE_KIND
                                                  DEFINE EDGE EXPLAIN SENT ×)
                          , 'IS TRUE SO' , 'TRUE' ) SET_EXPL_EDGE (× ENCES
                                                        ; ( 'BECAUSE ITS TRUE THE'
                             , 'IS NOT TRUE SO' , 'FALSE' ) SET_EXPL_EDGE
                                                ; ( 'BECAUSE ITS NOT TRUE THE'
                                                  DEFINE ALL STATUS AND IT ×)
    CAN THE CAR' , 'SET_ON' ) SET_NODE (× CAN THE CAR S FRAMES AND NODES'
                                        , 'DO_START' ) SET_NODE ; ( FALSE
                                      , 'TICK_START' ) SET_NODE ; ( FALSE
    CAN THE STARTER TAKE A TICK' , '? TAKE START , '? CAN THE BATTERY TAKE
                                       ON' , 'LOOSE' ) SET_NODE ; ( FALSE
                                      CAN' , 'LIGHT' ) SET_NODE ; ( FALSE

    FALSE , '? THE GRATE LIGHT IS HIGH , '? HEAD IS LOOSE , '? SOUND  ; (
                                                                    TRUE
    ) SET_NODE , '. KEEP THE BATTERY HEAD' , 'KEEP' ) SET_NODE ; ( ,
                                        'WIRE_REP' ) SET_NODE ; ( FALSE

    , '.REPAIRE OR CHANGE THE WIRES' , '? WIRES IS TRUE' , 'WIRE' OF+ TAKE
A CONNECTION FROM' , 'CONNECT' ) SET_NODE ; ( TRUE  ) SET_NODE ; ( FALSE
                                                    , '?CAN START. RTER

            , '?TEST THE SWITCH.ITS TRUE' , 'SWITCH_TEST' BATTERY TO STA
     TRUE , '.REPAIR THE SWITCH' , 'SWITCH_REP' ) SET_NODE ; ( REPAIR OR
CHANGE THE FALSE' , 'START_AUTOMAT' ) SET_NODE ; ( , 'OPEN_START' )
                                                        SET_NODE ; ( TRUE
                                                                 ; ( TRUE
                  , '.OPEN_AND REPAIR THE START' , 'AUTOMAT OF START  TRUE
    ; ( , '.CHECK OTHER SYSTEMS' , 'OTHER_SYSTEM' ) CHECK THE BODY
CONNECT.IS  IT  SET_NODE'  ,  'BODY_CNCT'  )  REPAIR  THE  SET_NODE'  ,
                                      'BODY_CNCT_REP' ) SET_NODE ; ( FALSE
                                           DEFINE ALL EDGE AND × ; ( TRUE
                                        PATH , '.BODY_CONNECT , '?TRUE)
            ; ( 0 , 'TRUE' , 'OTHER_SYSTEM' , 'SET_ON' ) DEFINE_EDGE (× S
                    ; ( 0 , 'FALSE' , 'DO_START' , 'SET_ON' ) DEFINE_EDGE
                ; ( 0 , 'TRUE' , 'OTHER_SYSTEM' , 'DO_START' ) DEFINE_EDGE
                 ; ( 0 , 'FALSE' , 'TICK_START' , 'DO_START' ) DEFINE_EDGE
                     ; ( 0 , 'TRUE' , 'LIGHT' , 'TICK_START' ) DEFINE_EDGE
                   ; ( 0 , 'FALSE' , 'LOOSE' , 'TICK_START' ) DEFINE_EDGE
                          ; ( 0 , 'TRUE' , 'KEEP' , 'LOOSE' ) DEFINE_EDGE
                        ; ( 0 , 'FALSE' , 'WIRE' , 'LOOSE' ) DEFINE_EDGE
```

```
                    ; ( 0 , 'TRUE' , 'CONNECT' , 'LIGHT' ) DEFINE_EDGE
                     ; ( 0 , 'FALSE' , 'LOOSE' , 'LIGHT' ) DEFINE_EDGE
                       ; ( 0 , 'TRUE' , 'LIGHT' , 'WIRE' ) DEFINE_EDGE
                   ; ( 0 , 'FALSE' , 'WIRE_REP' , 'WIRE' ) DEFINE_EDGE
               ; ( 0 , 'TRUE' , 'SWITCH_TEST' , 'CONNECT' ) DEFINE_EDGE
                ; ( 0 , 'FALSE' , 'BODY_CNCT' , 'CONNECT' ) DEFINE_EDGE
           ; ( 0 , 'TRUE' , 'SWITCH_REP' , 'SWITCH_TEST' ) DEFINE_EDGE
 ( 0 , 'FALSE' , 'START_AUTOMAT' , 'SWITCH_TEST' ) DEFINE_EDGE ; ( 0 ,
                    'TRUE' , 'OPEN_START' , 'BODY_CNCT' ) DEFINE_EDGE
                                                                    ×
; ( 0 , 'FALSE' , 'BODY_CNCT_REP' , 'BODY_CNCT' ) DEFINE_EDGE ; (×
                                                      CLOSING SYSTEM)
                                                ; SET_SN_ES_SYSTEM_OFF
                                                        ; ( FF ) CLOSE
                                                            ; END


                                              ; PROCEDURE FIRST_LOADING

   THIS PROCEDURE MAKE A KNOLEDGE BASE AND THEN LOAD IT × WITH DEFINE)
                                                          (× KNOLEDGE

                                                               BEGIN
                                                ; ( KB_NAME ) DEFINE_SN_ES
                                                          ; LOADING
                                                             ; END


                    ; BYTE : ( NAMES : NODE ) FUNCTION TRAVEL_STRATEGY

                                THIS PROCEDURE IS MAIN PART OF TRAVE ×)
                              IT RETURNED 4 DEFF - TROUGH THE STATES × (× LLING)
                                        1 FOR TRUE EDGE : VALUE × (× ERENT)
                                                                    »×
                                                  2 FOR FALSE EDGE ×)
                                                                    »×
                                               3 FOR RECIVING THE GOAL ×)
                                                                    »×
                                                 4 FOR ERROR IN ADDRES ×)
                                                                    (×

                                                                 VAR
                                                       ; BYTE : RESULT
                                                           ; CHAR : CH
                                                  ; LONGINT : ADDRES


                                                               BEGIN
 GETTING ADDRESS OF ×) ; ( NODE ) USER_ADDR =: ADDRES  1 THEN IN- <>
                                      IF ADDRES (× EXIST FRAME FILE ×) BEGIN
                                        {-I$} LOAD ×) ; ( ADDRES , FF ) SEEK
                                                    ; ( FRAME , FF ) IF READ
NOT FRAME.GOAL_NODE (× FRAME (× SO ASK F FRAME A REAL ADDRESS - NOT
                                          GOAL NODE ×) OR TRUE OR FALSE BEGIN
( '(YES OR NO )' , FRAME.MATTER_TEXT ) WRITELN (× THEN EDGE  REPEAT  ;
                              ( READKEY ) UPCASE =: CH ) OR ( 'Y' = CH ) UNTIL
                                  THEN 'Y' = IF CH (× TRUE EDGE ×) 1 =: RESULT
                                       (× FALSE EDGE ×) ; 2 =: ELSE END RESULT
```

```
                                                                ELSE END
×) ; 4 =: RESULT (× GOAL NODE ×) ; 3 =: ELSE RESULT ; ( 'N' = CH ; (×
                                                  RESULT =: TRAVEL_STRATEGY
                                                          ; END   PROCEDURE
: START_STATE ) PASSENGER ; ERROR IN ADDRES OF FRAME   × THIS PROCEDURE
                                                                    MAKE
                                        TRAVEL ; ( BOOLEAN : VAR ERROR ; NAMES)
                                BY USING THE TRAVEL STRATEG × (× BETWEEN STATES)
                                                                VAR (× Y
                                          (× CURRENT NODE NAME ×) ,CURRENT_NODE
                                (× NAME OF NEXT NODE OF THIS EDGE ×) , NODE_NAME
                                          (× KIND OF THIS EDGE ×) , EDGE_KIND
                                         (× ASKED EDGE KIND ×) ; NAMES : EDGE
                                       (× WEIGHT OF EDGE ×) ; INTEGER : WEIGHT
                                            (× FOUND THE ASKED EDGE ×) , FOUND
                                             (× FOUND THE GOAL ×) , GOAL_FOUND
                                   (× DIRECT OF THIS EDGE ×) ; BOOLEAN : DIRECT
                                           ACTION BY TRAVEL ×) ; BYTE : ACTION

                                                                      BEGIN
                                        IN ×) ; START_STATE =: CURRENT_NODE


RESET_PASS_NODES (× FIRST BEGIN AT START STATE (× STRATEGY  TION OF
                                                                      LOOP
=: ERROR (× RESET AL PASS NODE FLAG TO PREVEN SEARCH ×) ; REPEAT ;
                                                FALSE =: GOAL_FOUND =: ACTION
TAKE THE ×) ; ( CURRENT_NODE ) TRAVEL_STRATEGY ; CASE ACTION OF FALSE
                                          (× RATEGY ; 'TRUE' =: EDGE : 1 : 2
                                                ; TRUE =: GOAL_FOUND : 3 : 4
                                                              ; NOT END ) IF
( GOAL_FOUND ; TRUE =: ERROR ; 'FALSE' =: SERACH AND TRAVEL TRAVEL ST
                                                        EDGE ×) BEGIN  IF (×
×) THEN ( CURRENT_NODE ) THEN FIND_NODE ( NOT ERROR ) AND  ALLREADY ×)
                                                    ;TRUE =: ERROR (× ODE
                                               (× MAKE A LOOP - THIS NODE
        ) FIND THE CURRENT N PASS FROM FIND_FIRST_EDGE  =: FALSE FOUND
                                                                    ) WHILE
= SN_KB_ERROR ; ; ( DIRECT , WEIGHT , EDGE_KIND , NODE_NAME  LOOP ×)
                                                                      BEGIN

        DO FOR FOUND ASK ( NOT ERROR ) AND ( NOT FOUND ) AND ( BEGIN 0
FOUND ×) AND DIRECT THEN ( EDGE = EDGE_KIND ) IF (× ED EDGE   ( )
                                                        SIMPLE_EXPLAIN
                                                                      ×
; ( DIRECT ,CURRENT_NODE,NODE_NAME,EDGE × ASKED EDGE (× EXPLAIN MAKING)
                          MAKE A STATE TRAVE ×) ; NODE_NAME =: CURRENT_NODE
                  »; TRUE =: FOUND × (CHANGE NODE TO NEXT NODE) L STEP
                                                                    ; END
 , WEIGHT , EDGE_KIND , NODE_NAME ) FIND_NEXT_EDGE  ; NOT NOT FOUND
                                          THEN IF END ×) ; TRUE =: ERROR
                                        ; END (× UNTIL GOAL_FOUND OR GE
                                                        ; END   PROCEDURE
                                                         THIS PROCEDURE ×
MAKE A SEARCH IN ; USING ; EXIST ASKED ED ERROR ;( DIRECT (× STATE
                                                                SPACE)

                                                                    VAR
```

```
                                                    ; NAMES : START_STATE
                                                        ; BOOLEAN : ERROR

                                                                  BEGIN
                                                               ; CLRSCR
                                                  ; ( KB_NAME ) SET_SN_ES_SYSTEM_ON
                                                   ; ( 'FRM.'+KB_NAME , FF ) ASSIGN
                                                                       {-I$}
                                                            ; ( FF ) RESET
                                                         0 THEN <> IF IORESULT
                                                                ; EXIT
                                                  (× DEFINE START_STATE ×) REPEAT
                                                ; ( '? WHAT IS PROBLEM' ) WRITELN
                                                        ; ( START_STATE ) READLN
                                                   ; ( START_STATE ) FIND_NODE
                                                      0 THEN <> IF SN_KB_ERROR
                                  ; ( '.SORRY.NOT EXIST THIS PROBLEM IN KB' ) WRITELN
                                                    ; 0 = UNTIL SN_KB_ERROR
 CALL TRAVEL ×) ; ( ERROR , START_STATE ) PASSENGER  IF ERROR THEN
                                                    TATUS (× SPACE
                                                                  ELSE
) WRITELN ( 'ERROR IN KB' ) PASENGER ROTIN IN S WRITELN ; ( FF ) COPY
                                                  CLOSE ×) ; COPY_EXPLAIN
                                                                  (× T
EXPLAINS TO FILE EXPLAIN.TX ; ( FRAME.MATTER_TEXT  ; END  × (× MAIN)
                                                                  BEGIN
ONLY IN FIRST ×) ;FIRST_LOADING ; SET_SN_ES_SYSTEM_OFF (× IN ALL RUNNING
                                                  ×) ;USING  .END  %
 RUNNING ×)
```

ضمیمه 2 - نمونه محاورات‌انجام شده توسط برنامه سمپاد 1 .

تذکر :

نوشته های سمت راست از طرف سیستم و نوشته‌های سمت چپ از طرف
کاربر میباشد .

سلام - به‌سیستم سمپاد 1 خوش‌آمدید .
روی چه‌پایگاهی میخواهید کار کنید ؟
داده ها
چنین پایگاهی موجود نیست.آنرا بوجود بیاورم؟
بله
پایگاه ایجاد شد .
پایگاه بارگیری شد .

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :
کارمند

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :
علی

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :
محمد

موجودیت‌جدید ایجاد کنید ایجادموجودیت‌جدید.نام موجودیت‌را بنویسید :
مهدی

نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :
هست
نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :
فرزند
نوع ارتباط جدید ایجاد کنید
ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :
مرد
وضعیت‌ارتباط را تعیین کنید وضعیت‌چه‌ارتباطی‌تعیین‌شود؟
هست آیاارتباط توارث‌مستقیم‌دارد؟
بله
آیاارتباط توارث‌معکوس‌دارد؟
بله
آیاارتباط یکتائی مستقیم‌دارد؟
خیر آیاارتباط یکتائی معکوس‌دارد؟
خیر
وضعیت‌ارتباط را چاپ‌کنید وضعیت‌چه‌ارتباطی‌چاپ‌شود؟
هست توارث‌معکوس_ توارث‌مستقیم _
وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
هست توضیح ارتباط درحالت مستقیم چیست؟
هست‌یک توضیح ارتباط درحالت معکوس چیست؟
وجود دارد به نام

وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
فرزند توضیح ارتباط درحالت مستقیم چیست؟
فرزند کسی است‌به نام
توضیح ارتباط درحالت معکوس چیست؟
فرزندی دارد به نام

وضعیت‌توضیح ارتباط را تعیین کنید وضعیت توضیح چه ارتباطی تعیین شود؟
مرد توضیح ارتباط درحالت مستقیم چیست؟
مردی است‌با نام

توضیح ارتباط درحالت معکوس چیست؟
مردی است‌با نام

وضعیت‌توضیح ارتباط را چاپ‌کنید توضیح چه ارتباطی چاپ شود؟
فرزند معکوس:فرزندی دارد به نام مستقیم:فرزند کسی است‌به نام

ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
علی
به چه موجودیتی ارتباط برقرار شود ؟
کارمند چه نوع ارتباطی‌برقرار شود؟
هست باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (ESC
ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
محمد به چه موجودیتی ارتباط برقرار شود ؟
مهدی
چه نوع ارتباطی‌برقرار شود؟
فرزند باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (ESC
ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
علی
به چه موجودیتی ارتباط برقرار شود ؟
محمد چه نوع ارتباطی‌برقرار شود؟
فرزند باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (ESC
ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
کارمند به چه موجودیتی ارتباط برقرار شود ؟

ارتباط جدید ایجاد کنید
از چه موجودیتی ارتباط برقرار شود ؟
علی
به چه موجودیتی ارتباط برقرار شود ؟
کارمند چه نوع ارتباطی‌برقرار شود؟
هست باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (ESC وجود داشته .

ارتباط جدید ایجاد کنید از چه موجودیتی ارتباط برقرار شود ؟
مهدی
به چه موجودیتی ارتباط برقرار شود ؟
مهدی
چه نوع ارتباطی‌برقرار شود؟
مرد باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (ESC

نوع ارتباط جدید ایجاد کنید ایجادنوع‌ارتباط جدید. نوع‌ارتباط را بنویسید :
پدربزرگ
ارتباط مرکب‌ایجاد کنید کدام ارتباط به‌ارتباط مرکب تبدیل شود ؟
پدربزرگ بترتیب ارتباط توالی را ازآخر به‌اول‌بنویسیدودرانتهای‌توالی‌باESCخارج‌شوید. ارتباط ساده توالی را بنویسید :
مرد آیا ارتباط درحالت مستقیم است ؟
بله
ارتباط ساده توالی را بنویسید :
فرزند آیا ارتباط درحالت مستقیم است ؟
خیر ارتباط ساده توالی را بنویسید :
فرزند آیا ارتباط درحالت مستقیم است ؟
خیر ارتباط ساده توالی را بنویسید :
ESC
خداحافظ
به همین‌زودی خسته شدید ؟ یا کارتان تمام‌شد ؟
بله

به امید دیدار مجدد .

سلام - به‌سیستم سمپاد _1 خوش‌آمدید .
روی چه‌پایگاهی میخواهید کار کنید ؟
داده ها
پایگاه بارگیری شد .

وضعیت‌توضیح ارتباط را چاپ‌کنید توضیح چه ارتباطی چاپ شود؟
پدربزرگ معکوس:پدربزرگی دارد به نام مستقیم:پدر بزرگ‌کسی است‌به نام
جستجو کنید
از چه‌موجودیتی جستجو کنم؟
کارمند چه‌نوع ارتباطی را بیابم؟
پدربزرگ آیا ارتباط درحالت مستقیم جستجو شود ؟
خیر
آیا توارث درحالت مستقیم تعیین شود ؟
خیر باچه‌وزنی ارتباط برقرار شود؟) اگر وزن اهمیتی ندارد ESC کنید (
ESC
مهدی

توضیح دهید به دلیل اینکه کارمند وجود دارد به نام علی پس
یک پدربزرگ مردی است‌با نام فرزندی دارد به نام فرزندی دارد به نام
آن‌است
علی فرزند کسی است‌به نام محمد محمد فرزند کسی است‌به نام مهدی مهدی مردی است‌با نام مهدی علی پدربزرگی دارد به نام مهدی
خداحافظ

به همین‌زودی خسته شدید ؟ یا کارتان تمام‌شد ؟
بله

به امید دیدار مجدد .