NTNU
Kunnskap for en bedre verden

Institutt for datateknikk og informasjonsvitenskap

Eksamensoppgave i TDT4137 Kognitive Arkitekturer

Faglig kontakt under eksamen: Asbjørn Thomassen

Tlf.: 73591839

Eksamensdato: 5 desember 2016

Eksamenstid (fra-til): 0900-1300

Hjelpemiddelkode/Tillatte hjelpemidler: D/ Kalkulator

Annen informasjon:

Målform/språk: bokmål

Antall sider (uten forside): 5

Antall sider vedlegg: 1

Informasjon om trykking av eksamensoppgave

Originalen er:

1-sidig ☐ **2-sidig** ☐

sort/hvit ☐ **farger** ☐

skal ha flervalgskjema ☐

Kontrollert av:

28/11-2016
Dato Sign

Merk! Studenter finner sensur i Studentweb. Har du spørsmål om din sensur må du kontakte instituttet ditt. Eksamenskontoret vil ikke kunne svare på slike spørsmål.

Oppgave 1 (20%)

a) Vi har 4 viktige aspekter i modelleringen av kognitive systemer (Vernon). Redegjør kort for disse. **Svar:**

1. The computational / bio-inspired spectrum (hvorfra hentes inspirasjon)
2. The level of abstraction in the biological model
3. The mutual dependence of brain, body, and environment.
4. The ultimate-proximate distinction (why-how vurdering)

b) Hva er påstandene som settes frem i : "The Physical Symbol System Hypothesis " og "The Heuristic Search Hypothesis"?
Svar:
"The Physical Symbol System Hypothesis ": A physical symbol system has the necessary and sufficient means for general intelligent action.

"The Heuristic Search Hypothesis": The solutions to problems are represented as symbol structures. A physical-symbol system exercises its intelligence in problem- solving by search, that is, by generating and progressively modifying symbol structures until it produces a solution structure

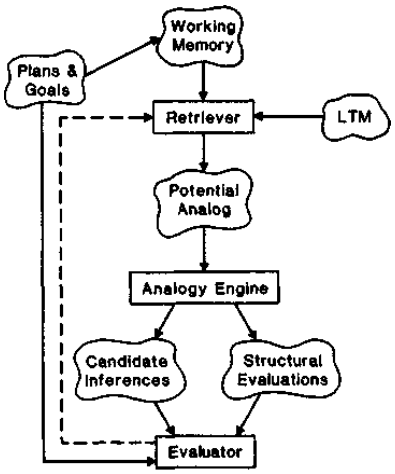
c) Definer ulike typer av similaritet relatert til domenekunnskap representert som objekter, attributter og relasjoner (Gentner). **Svar:**

Kinds of Predicates Mapped in Different Types of Domain Comparison

	No. of attributes mapped to target	*No. of relations mapped to target*	*Example*
Literal Similarity	Many	Many	The K5 solar system is like our solar system.
Analogy	Few	Many	The atom is like our solar system.
Abstraction	Few[a]	Many	The atom is a central force system.
Anomaly	Few	Few	Coffee is like the solar system

[a]Abstraction differs from analogy and the other comparisons in having few object-attributes in the *base* domain as well as few object-attributes in the target domain.

d) Figuren nedenfor viser en arkitektur for analogi-resonnering:



Forklar hvordan "Analogy Engine" virker.

Svar: Tre trinn: (tegn gjerne opp som artikkelen)

1. **Mapping the target and the base candidate** (blind and local by matching all indentical predicates and subpredicates)
2. **Construct similarities** (local matces are coalesced into structurally consistent connected clusters, called **kernals**, by enforcing one-to- one mapping and parallell connectivity between them).
3. **Identify inferences** (Merge kernals into one or few maximal structurally consistent interpretations for analogy (mapping one-to- one mapping and parallell connectivity). Then do **structure evaluation** of interpretations (favors deep Return best match and others within 10% of it.

e) Redegjør kort for MAC/FAC-implementasjonen og hvordan den finner analogier.

Svar: Implementation of Gentner's approach to analogy:

Two stages of 'Many Are Called but Few Are Chosen':

a. MAC: retrieval of candidate 'bases'. Uses 'content vectors' (CV)
   uses both entity, attribute, relations, functions, i.e., it employs '**literal similarity**'

b. FAC: selection of *the* base. Uses 'structure mapping engine' (SME)
   elaborates how to map the target to the base
   has an explanatory nature.

Oppgave 2 (20%)

Du skal i denne oppgaven modellere, i NGOMSL, en del av det å lage trekanter i Powerpoint. Oppgaven er begrenset til å tegne en enkel trekant basert på **kopier-og-lim-inn**. Dette kan gjøres på to måter:

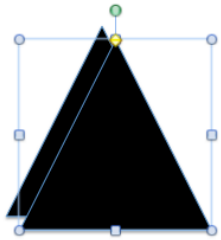
Alternativ 1: Du **markerer** først en trekant du allerede har på skjermen.

En markering kan enten gjøres ved å flytte kursor innenfor trekanten og gjøre et museklikk, eller ved å bruke et markerings-rektangel, se Figur 1.
Markerings-rektangelet kommer frem ved at du først trykker ned muse-knappen i en startposisjon **utenfor** objektet (trekanten). Et markeringsrektangel med ankerpunkt i startposisjonen vil nå følge kursor som vist i Figur 1. Når trekanten er helt dekket av rektangelet og museknappen slippes, vil den bli markert som vist i Figur 2.

Figur 1. Bruk av markerings rektangel for å velge trekanten

Figur 2. Markert triangel klart for kopiering

Når trekanten er markert kan du gjøre **kopier-og-lim-inn** ved å trykke ned **cmd**-tasten sammen med **c** (copy) og deretter sammen med **v** (paste).
Merk at den nye trekanten er markert, legger seg rett over den første trekanten og må dras ned til ønsket posisjon:



Alternativ 2: Du kan holde Option-tasten nede og trykke ned museknappen over en allerede eksisterende trekant. En markert kopi av trekanten vil nå feste seg til kursor og kan dras til ønsket posisjon hvor museknappen slippes.

a) Modeller målet *createTriangleByCopyAndPaste* i NGOMSL. Du skal altså her ikke ta med unit-task-nivået og heller ikke modellere bruk av WM/LTM.
Svar:
(Her ser vi etter at kandidaten har brukt **selection rule set** for å modellere valg mellom alternativer. Har **to** seleksjonsregler: en for alternative måter å kopiere trekant og en for to måter å markere en trekant). Metoder utfører mål/submål ved å utføre operatorer i steps (mål/submål har rød farge i koden under). Løsningsforslag mental program i NGOMSL:

```
Selection rule set for the goal: createObjectByCopyAndPaste
If task is create rectangle by copy and paste then accomplish:
objectCopyAndPaste
If task is  create rectangle by option-key and draging : copyObjectByOptionDrag
Return with goal accomplished


Method for goal: copyObjectByOptionDrag
Step 1. Locate source Object
Step 2. Move cursor to Object
Step 3. Press Option-key   // key must be down until mouse release
Step 4. Accomplish goal: Drag Object to destination
Step 5  Release Option-key
Step 6. Return with goal accomplished


Method for goal: drag object to destination
Step 1. Press mouse button on object
Step 2. Verify object inverse
Step 3. Locate destination
Step 4. Move object to destination
```

```
Step 5. Release mouse
Step 6. Return with goal accomplished


Method for goal: objectCopyAndPaste
Step 1. Accomplish goal: mark object to copy
Step 2. Press cmd-C  // do copy
Step 3. Press cmd-V  //paste
Step 4. Decide: If cursor not inside newObject then Move cursor to newObject
Step 5. Drag newObject to wanted position // uses generic method " drag object
to destination"
Step 6. Return with goal accomplished



Selection rule for goal: mark Object
If task is  mark by click: markObjectbyClick
If task is mark by using rectangle: markObjectbyRectangle
Return with goal accomplished

Method for goal: markObjectbyClick
Step 1. Locate Object
Step 2. Move cursor to Object
Step 3. Press Option-key
Step 4. Click
Step 5. Verify
Step 6. Return with goal accomplished

Method for goal: markObjectbybyRectangle
Step 1. Locate Object
Step 2. Move cursor below and left of Object   //many variants
Step 3. Press mouse button
Step 4. Move cursor so that Object is inside marque rectangle
Step 5. Release mouse
Step 5. Verify that Object is marked
Step 6. Return with goal accomplished
```

b) Beregn utførselstid (execution time) for å lage en trekant i a) ved bruk av markeringsrektangel.
(CP = 1.2, B=0.1, H=0.4, K=0.2, M=1.2, P = 1.1 - alt i sekunder)

Svar: Her skal KLM-operatorerer som er brukt telles opp (CP:cogntive_perceptual action, B: button up/down, K_ keystroke, P: point (move) , H: hands home). Så skal det legges til mental adminstrasjonskostnad på 0.1 sek pr. step utført, dvs tiden beregnes etter formel:

$$\mathbf{T} = 0.1*\text{no\_steps} + \Sigma\ \text{KLM\_operators}$$

c) Konsistens kan defineres som gjenbruk av prosedural kunnskap. Hvordan avspeiles dette i målgrafen og NGOMSL-modellen?

Svar: Gjenbruk av sub-mål og metoder.

Gi eksempler på hvordan du kan parallellisere operatorer i modellen din?

Svar: Locate and move, verify and move, etc

Hvordan er dette relatert til den underliggende kognitive arkitekturen (MHP)?

Svar: Uavhengige subsystemer (F.eks perseptuelt og motorsystem kan kjøre i parallell)

Hva er forskjellen på if-setninger i seleksjonsregler og metoder?

Svar: If-setninger i seleksjonsregler utføres i parallell (produksjonssystem if). If i metoder er sekvensiell ("vanlig" if fra programmering)

Hvilke operatorer har vi for lagring og gjenfinning av informasjon (memory storage and retrieval)?

Svar: recall-WM, retain-WM, retrieveL-TM, forget-WM

Oppgave 3 (20%)

a) Hva er et rekurrent nevralt nett? **Svar:** har tilbakekoblinger/sykler.
Hvordan trener du en autoenkoder (supervised learning)? Svar: input settes lik output. Identitetsfunksjon.
Hva menes med dype nevrale nett? **Svar:** Har mange (skjulte) lag.
Hvilken klasse av kognitive systemer hører nevrale nett under? **Svar:** Emergente/konneksjonisme
Hva er den overordnede ideen bak deltaregelen? **Svar:** Gjøre gradient-nedstigning i kostnadsfunskjonen (feilen). Går små skritt mot et (forhåpentligvis) globalt mininum. Kan bli sittende fast i lokale minima.

b) Vi skal trene opp et feed-forward-nettverk vha backpropagation-algoritmen. Deltareglen for læring i output-laget er gitt ved:

$$\Delta w_{jk}(p) = \alpha\, y_j(p)\, \delta_k(p)$$
$$\delta_k(p) = y_k(p)\,(1 - y_k(p))\, e_k(p)$$

og for hidden-laget:

$$\Delta w_{ij}(p) = \alpha\, x_i(p)\, \delta_j(p)$$
$$\delta_j(p) = y_j(p)\,(1 - y_j(p)) \sum_k \delta_k(p)\, w_{jk}(p)$$

Aktiveringsfunksjonen er gitt ved: k

$$y_j = \frac{1}{1+e^{-X_j}}$$

hvor X_j er netto input til nevronet.

Hva er α? Definer X_j og e_k.
Svar:

α er læringshastigheten. $e_k = y_d - y$ (d for desired output)

Net weighted input to neuron:

$$X_j = \sum_{i=1}^{k} x_i w_{ij} - \theta_j = \sum_{i=0}^{k} x_i w_{ij}$$

Beskriv trinnene i backpropagation.
Svar:

1. Initialisation:

set weights and thresholds to random numbers uniformly distributed *[-2.4/F_i,0.5/F_i]* where F_i is number of inputs to neuron i.

2. Activation:

Calculate for neuron j in hidden layer:

$$y_j(p) = sigmoid\left[\sum_{i=1}^{n} x_i(p)w_{ij}(p) - \theta_j\right]$$

Calculate for neuron k in output layer:

$$y_k(p) = sigmoid\left[\sum_{j=1}^{m} x_{jk}(p)w_{jk}(p) - \theta_k\right]$$

3. Weight training:

Calculate gradient for neurons output layer:

$$\Delta w_{jk}(p) = \alpha\ y_j(p)\ \delta_k(p)$$
$$\delta_k = y_k(p)\ (1- y_k(p))\ e_k(p)$$

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p)$$

Error gradient for neurons hidden layer:

$$\Delta w_{ij}(p) = \alpha\ x_i(p)\ \delta_j(p)$$

$$\delta_j = y_j(p)\ (1- y_j(p))\sum_{k=1}^{l} \delta_k(p)\ w_{jk}(p)$$

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

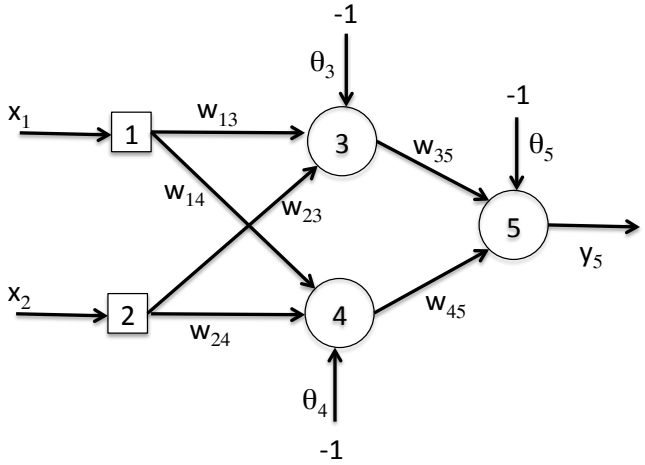
4. Iteration: p = p +1, goto 2 until error criterion is satisfied

NB! θ_j sees på som en vekt w_{0j} koblet til en input -1 og regnes ut som en vanlig w.

c) Gitt nettverket under med følgende verdier etter iterasjon p:

$x_1=1,\ \ x_2=0,\ \ y_d=1,\ \ \alpha=0.1$

$w_{13}= 0.5, w_{14}=0.9,\ w_{23}=0.4,\ w_{24}=1.0, w_{35}=-1.2,\ w_{45}=1.1,\ \theta_3=0.8,\ \theta_4=-0.1, \theta_5 = 0.3$



Vis hvordan du bruker algoritmen i b) til å beregne endringene i vekter. Du skal her bare gjøre en iterasjon (fra p til p+1)

Svar: Viktig her å bruke algoritmen riktig: feilen forplantes bakover i step 3. Det er også lett å glemme w_{0j} , i.e θ_j , som må endres på lik linje som vanlige vekter w.

Må først regne ut output y5:

y3=0.4256, y4 = 0.7311 som gir **y5** = 0.4984 (step 2 i algoritmen)

Så må en gjøre backpropagation ved først å regne endringer i output layer (step3): Setter bare inn i oppgitte formler i oppgaven:

Finner først $e = y_d - y_5 = 0.5016$ som gir
$\delta_5 = 0{,}1254$ og dermed
$\Delta w_{35} = 0{,}0053$
$\Delta w_{45} = 0{,}0092$
$\Delta\theta_5 = -0{,}0125$ kan sees på som Δw_{05}

Nå kan en gå til hidden layer:

$\delta_3 = -0{,}0368$ og $\delta_4 = 0{,}0271$ dermed
$\Delta w_{13} = -0{,}0037$
$\Delta w_{14} = 0{,}0027$
$\Delta\theta_3 = 0{,}0037$
$\Delta w_{23} = 0{,}0$
$\Delta w_{24} = 0{,}0$
$\Delta\theta_4 = -0{,}0027$

Disse endringene legges så til gamle verdier av vektene….

Oppgave 4 (20%)

Du er på restaurant med venner og skal betale for maten. Men hva skal du gi i tips? For å løse problemet har du heldigvis installert en ”intelligent tipping agent” på telefonen din. Agenten bruker fuzzy-resonnering med to input-variable: `service` og `food`. Variabelen `service` opererer på settene: **Poor**, **Good** og **Execellent**, og `food` har fire sett: **Tasteless**, **Average**, **Delicious** og **Gourmet**. Output-variabelen `tip` har settene: **Cheap**, **Average** og **Generous**.

Alle settene med regler er beskrevet i Vedlegg 1.

a) Gitt reglene og inn-verdier for `service` = 4 og `food` = 8.
Hva anbefaler agenten i tips?
Anta Mamdani-resonnering og vis trinnene i hvordan du kommer frem til svaret.

Du kan skrive og tegne direkte på Vedlegg 1. For enkelthets skyld kan du bruke 3 som step-størrelse i integreringen.

Svar:

1. Fuzzification (gir verdiene som vist i fuzzy settene i vedlegg 1)
2. Rule evaluation (gir verdier som vist over reglene i vedlegg 1)
3. Aggregation (se svart sett tegnet inn på action-settet i vedlegg 1)
4. Defuzzication. Mamdani bruker Centre of gravity (COG):

$$COG = \frac{\int_a^b \mu_A(x)\, x\, dx}{\int_a^b \mu_A(x)\, dx}$$

– tilnærmer integrasjonen med step på 3 fra 0 til 30 langs x (finner areal under svart kurve i tip-settet). Agenten anbefaler:
= 0.0*(0+3)+0.2*6+0.5*9+0.6*(12+15+18)+0.5*21+0.3*(24+27+30) / (0.0*2+0.2+0.5+ 3*0.6+0.5+3*0.3) = 67.5/3.9 = **ca 17.3**

b) Hva skiller Mamdani fra Sugeno?

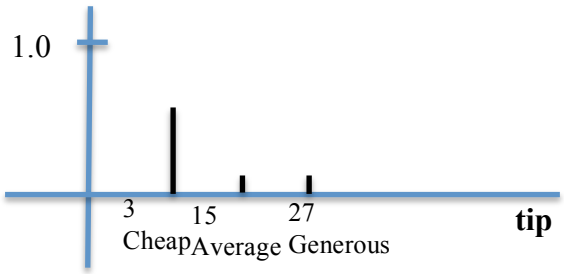
Svar:
Mamdani er lettere å forstå for mennesket, men tyngre å beregne (COG). For Sugeno er det motsatt (spesielt lett å beregne).

Skriv om reglene i Vedlegg 1 til Sugeno ved at du velger singeltons i 3, 15 og 27 for `tip`.

Svar:
If (service is Poor) OR (Food is Tasteless) then (tip is 3)
If (service is Good) then (tip is 15)
If (service is Excellent) AND (food is Delicious OR food is Gourmet) then (tip is 27)

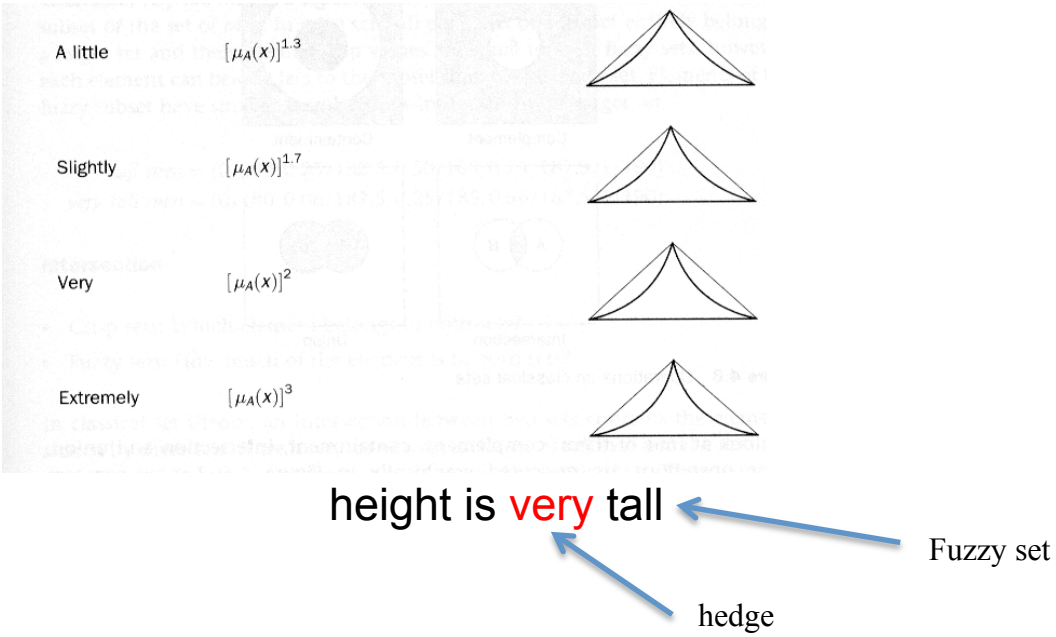
Vis hvordan du nå beregner tips.

Svar: Oppgaven gir singeltons i 3,15 og 27 (erstatter Mamdani-fuzzysettene Cheap, Average og Generous).
Singeltons under blir altså klippet ned til hhv 0.0, 0.6 og 0.3 ved evaluering av reglene.



Agenten anbefaler med Sugeno(vekta sum av klippet singeltons)
= (0.0*3+0.6*15+0.3*27)/(0.0+0.6+0.3)=17.1/0.9 = **ca 19 (mellom Average og Generous)**

Hvordan bruker man en `hedge` i Fuzzy-resonnering?
Svar: En hedge er modifikator (funksjon som avvendes) på et fuzzy set:



Oppgave 5 (20%)

a) Hvorfor kan en klassifisere ACT-R som et kognitivt hybrid system?
Svar: Har et både et symbolsk og subsymbolsk subsystem (men ok om inspirasjon fra neuro-vitenskap, psykologi og AI)

b) Basis-nivå-aktivering B_i av en chunk i ACT-R er definert ved:

$$B_i = \ln\left(\sum_j t_j^{-d}\right)$$

Hva utrykker denne ligningen?
Svar: Base-Level activation reflects the log-odds that a chunk will be needed. In the environment the odds that a fact will be needed decays as a power function of how long it has been since it has been used. The effects of multiple uses sum in determining the odds of being used.

c) Koden i Soar nedenfor er fra det kjente "Water-jug"-problemet, hvor vi har to mugger med kapasitet på 3 og 5 liter. Attributten `^contents` er vannkapasiteten i liter og `^empty` angir ledig plass i en mugge.

```
sp {water-jug*propose*pour
   (state <s> ^name water-jug
              ^jug <i> { <> <i> <j> })
   (<i> ^contents > 0 )
   (<j> ^empty > 0)
-->
   (<s> ^operator <o> + =)
```

```
(<o> ^name pour
     ^empty-jug <i>
     ^fill-jug <j>)}
```

Hva gjør koden? Hvorfor **utføres** ikke pour her?

Svar: Koden bare **foreslår** en operator med preferanse "Acceptable" som tømmer vann fra mugge <i> til <j> **dersom** det finnes to mugger <i> og <j> hvor `^contents >0 og ^empty > 0.` Dvs. den ene muggen må ha innhold som kan tømmes og den andre ha plass. Operatorforslaget kobles til tilstanden <s> og har navn og parametre . Soar kan da se litt på konskvensene av forslaget (i elaborerings-fasen, se d)) før Soar evt. velger å utføre den. Valg av operator skjer i Soars decision-fase som velger kun **en** operator. Først når Soar har valgt operatoren, utføres den ved en bruk av en application-rule.

Hva betyr det å fjerne `+ =` fra kodelinja `(<s> ^operator <o> + =)`?
Svar: Fjerning av += betyr at vi tester på om det finnes en Soar-valgt operator (lagt ut av Soar i Decision-fasen) som skal utføres. Soar legger altså ut `(<s> ^operator <o>)` for å signalisere at operator er valgt i tilstanden <s>.

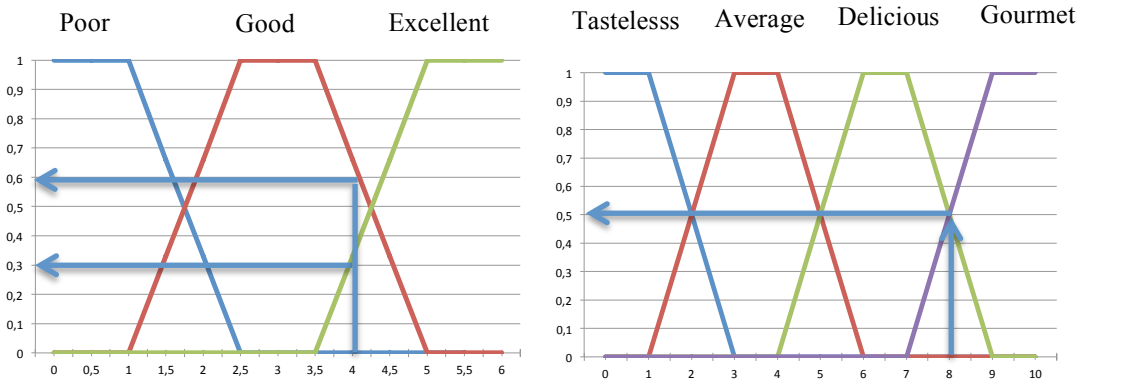
d) Hva hensikten med elaboreringsfasen i Soar?
Svar: Skaffe tilveie mest mulig kunnskap i situasjonen og å foreslå/evaluere ulike forslag til operatorer. Regler fyrer i faser til "quiesence" før decision-fasen utføres av Soar.

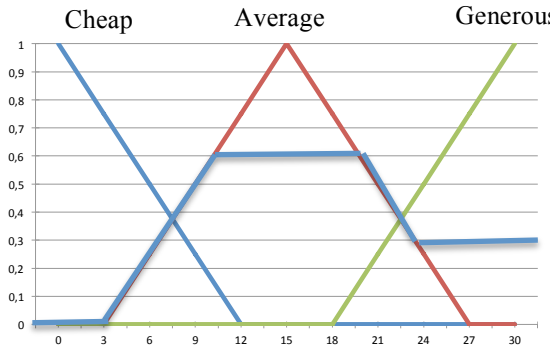
Anta at **alle** operatorer har fått lik preferanse "+ ". Hva gjør Soar før å løse situasjonen?
Svar: Det har oppstått en *impasse*. "A *substate* is automaticly created that represents the information relevant to resolving the impasse (including the original state)".
Soar generer en nytt problemrom/subtilstand for å prøve å skaffe til veie kunnskap om hvilken operator den skal velge, dvs løse operatorvalget ved å endre operator-preferanser (Ved preferanse += velger Soar random blant operatorene).
Kan også skrive litt om typer av impasse ++ om bruk av semantic og episodic memory

Fuzzy-sett og regler:

Svar: leser av verdier på grafer ved 4 og 8

OR = Max og AND = min av verdier over reglenes venstre side. Overfører verdi til høyre side





Klipper output sett med verdiene fra reglene nedenfor (høyre side) – gir nytt aggregert sett

0.0 0.5 =>**0.0**

If (service is Poor) OR (Food is Tasteless) then (tip is Cheap)

0.6 =>**0.6**

If (service is Good) then (tip is Average)

0.3 0.5 0.5

If (service is Excellent) AND (food is Delicious OR food is Gourmet) then

=>0.3

(tip is Generous)