Pràctica de Sistemes Basats en el Coneixement

Sistema Expert de Recomanació d'Habitatges de Lloguer

Grau IA – Q1 Curs 2025-2026

Departament de Ciències de la Computació
Universitat Politècnica de Catalunya

Anel Ademovic
anel.suljic@estudiantat.upc.edu

Aleix Pitarch
aleix.pitarch@estudiantat.upc.edu

Joan Solina
joan.solina@estudiantat.upc.edu

14 de desembre de 2025

Índex

1 Identificació del problema

1.1 Descripció del problema

El mercat immobiliari de lloguer a Barcelona presenta una complexitat creixent que dificulta la cerca d'habitatge adequat per part dels ciutadans. La regidoria d'habitatge de l'ajuntament de Barcelona disposa d'un gran nombre d'ofertes de lloguer, però connectar aquestes ofertes amb les persones que busquen habitatge de manera eficient requereix un coneixement expert que tingui en compte múltiples factors simultàniament.

El problema que ens proposem resoldre va més enllà d'una simple cerca parametritzada. No es tracta només de filtrar ofertes per preu o nombre d'habitacions, sinó d'entendre les necessitats reals de cada tipus de solůlicitant i fer recomanacions intelůligents que considerin el context complet de la seva situació vital. Per exemple, una família amb fills petits no només necessita un pis amb suficients habitacions, sinó que valora especialment la proximitat a escoles, zones verdes i serveis pediàtrics, mentre que evita zones amb soroll excessiu. Una persona gran, per contra, prioritza l'accessibilitat, la proximitat a centres de salut i comerços de proximitat. Aquestes preferències no sempre són explícites en la cerca inicial del solůlicitant, però un expert immobiliari les tindria en compte.

A més, el sistema ha d'anar més enllà de les característiques intrínseques de l'habitatge (superfície, nombre d'habitacions, preu) i considerar el seu entorn urbà. La localització de l'habitatge en relació amb serveis com transport públic, comerços, centres educatius, sanitaris i zones d'oci és crucial per determinar la seva adequació. Aquest coneixement territorial no és trivial: cal saber què significa estar "a prop"d'un servei segons el context (una persona gran considera prop el que està a 200m, mentre que un jove pot considerar acceptable 500m), i entendre quins serveis són molestos per a qui (una discoteca propera pot ser desitjable per a estudiants però inacceptable per a famílies amb nens petits).

El repte també inclou gestionar restriccions de diferent naturalesa. Algunes són absolutes (permetre mascotes quan el solůlicitant en té, accessibilitat per a persones amb mobilitat reduïda), altres són preferències fortes (pressupost màxim), i d'altres són desitjables però no imprescindibles (orientació solar, vistes). El sistema ha de ser capaç de distingir entre aquests nivells i generar recomanacions que s'adaptin al grau de flexibilitat de cada solůlicitant.

Finalment, el sistema no només ha de trobar habitatges que compleixin uns criteris mínims, sinó que ha de classificar-los en diferents graus de recomanació: des d'ofertes parcialment adequades (que compleixen la majoria de requisits però fallen en algun aspecte menor) fins a ofertes molt recomanables (que no només compleixen tots els requisits sinó que ofereixen avantatges addicionals). Aquesta classificació ha de venir acompanyada d'explicacions clares que permetin al solůlicitant entendre per què una oferta està recomanada i què la fa destacar o quins aspectes caldria considerar abans de prendre una decisió.

1.2 Anàlisi de viabilitat

Per determinar si aquest problema és adequat per ser resolt mitjançant un sistema basat en el coneixement, hem analitzat diverses dimensions de viabilitat.

1.2.1 Viabilitat tècnica

El problema reuneix les característiques fonamentals que fan viable la construcció d'un SBC:

Existència de coneixement expert. Existeix un coneixement expert clar en el domini immobiliari. Els agents immobiliaris experimentats desenvolupen una comprensió profunda de com emparedar perfils de clients amb habitatges adequats, considerant factors que van més enllà de les especificacions tècniques. Aquest coneixement inclou heurístiques del tipus l·les famílies amb fills petits valoren especialment la proximitat a escoles i parcs"o l·les persones grans necessiten accessibilitat i serveis de salut propers", que es poden formalitzar en regles.

Domini delimitat. Hem acotat el problema a la ciutat de Barcelona, amb tipus d'habitatge i serveis ben definits. Aquesta delimitació fa el problema tractable sense perdre la seva essència. No intentem resoldre el problema general de recomanació d'habitatges a nivell mundial, sinó que ens centrem en un context urbà específic amb característiques ben conegudes.

Problema de classificació i recomanació. El problema s'ajusta perfectament a una metodologia de classificació heurística: hem de classificar ofertes en categories de recomanació (parcialment adequades, adequades, molt recomanables) basant-nos en l'avaluació de múltiples criteris. Aquest tipus de problema és un dels més adequats per SBC, ja que podem descompondre'l en subproblemes (abstracció del solůlicitant, càlcul de proximitats, descart d'ofertes, puntuació, classificació) que es resolen seqüencialment aplicant regles.

Espai de solucions finit. Tot i que l'espai de combinacions possibles entre solůlicitants i ofertes és gran, és finit i manejable. Amb N solůlicitants i M ofertes, tenim NŒM parells a avaluar, però cada avaluació és independent i es pot resoldre aplicant un conjunt de regles ben definides.

1.2.2 Viabilitat de desenvolupament

El projecte és viable des del punt de vista del desenvolupament per diverses raons:

Disponibilitat d'eines. CLIPS proporciona un entorn robust per implementar sistemes basats en regles, amb suport per a programació orientada a objectes (COOL) que ens permet representar l'ontologia del domini de manera natural. Protégé ens ha permès dissenyar i documentar l'ontologia de forma sistemàtica abans de la implementació.

Desenvolupament incremental. El problema es pot abordar de manera incremental, començant amb un conjunt bàsic de regles i ampliant progressivament la cobertura. Hem pogut començar amb la gestió de restriccions dures (preu, mascotes, accessibilitat) i anar afegint regles més sofisticades per a la puntuació i classificació.

Prototipatge ràpid. La naturalesa declarativa de CLIPS permet fer prototips ràpidament i iterar sobre el disseny. Modificar o afegir regles no requereix reescriure grans porcions de codi, cosa que facilita l'experimentació i refinament.

1.2.3 Limitacions identificades

Tot i la viabilitat general, hem identificat algunes limitacions que cal tenir presents:

Coneixement incomplet del domini. No som experts immobiliaris reals, per la qual cosa el nostre sistema es basa principalment en coneixement de sentit comú i en patrons generals que hem pogut inferir. Un sistema de producció requeriria la participació activa d'experts del sector per afinar les regles i els pesos de puntuació.

Dades sintètiques. Les instàncies que utilitzem són simulades. Un sistema real necessitaria integrar-se amb bases de dades reals d'ofertes i disposar d'informació actualitzada sobre serveis urbans. També caldria considerar la dinàmica temporal (ofertes que deixen d'estar disponibles, preus que canvien).

Absència de retroalimentació. El sistema actual no aprèn de les decisions dels usuaris. No sabem si les recomanacions que fem són realment útils o si els usuaris acaben escollint opcions diferents de les que el sistema proposa com a millors. Un sistema real hauria d'incorporar mecanismes de feedback per ajustar els pesos i les regles.

Factors subjectius. Alguns factors que influeixen en la decisió d'escollir un habitatge són altament subjectius i difícils de codificar en regles (l'estètica del barri, la "sensació"que transmet un habitatge, factors culturals o personals molt específics). El nostre sistema se centra en factors objectius o preferències generals, però no pot capturar aquestes nuances individuals.

1.3 Fonts de coneixement

Per construir el sistema hem identificat i utilitzat diverses fonts de coneixement:

1.3.1 Fonts primàries

Coneixement de sentit comú. La base principal del nostre sistema prové del sentit comú sobre necessitats habitacionals segons perfils demogràfics. Aquest coneixement inclou afirmacions com l·les persones grans necessiten accessibilitat"o "els estudiants valoren la proximitat al transport públic", que són àmpliament acceptades i no requereixen expertesa específica.

Normativa d'accessibilitat. Hem consultat els requisits bàsics d'accessibilitat (presència d'ascensor en plantes altes, accés sense barreres) que són estàndards regulats i objectius.

Webs d'anuncis immobiliaris. Plataformes com Idealista ens han servit per identificar les característiques rellevants que es descriuen en les ofertes reals (superfície, nombre d'habitacions, si permet mascotes, si té terrassa, consum energètic, etc.). També ens han permès veure quines són les categories de serveis que es mencionen habitualment com a punts forts de les ubicacions.

1.3.2 Fonts secundàries

Models de llenguatge (opcional). Tal com s'indica a l'enunciat, podríem haver utilitzat models de llenguatge com a "experts"per fer elicitació de coneixement sobre criteris de decisió en la recomanació d'habitatges. Tot i que no ho hem documentat extensament en aquest apartat, seria una via legítima per obtenir coneixement estructurat sobre el domini.

Experiència personal. Els membres de l'equip hem aplicat la nostra pròpia experiència en la cerca d'habitatge i coneixement de la ciutat de Barcelona per definir proximitats raonables, serveis rellevants i preferències típiques.

Ontologies existents. Hem consultat exemples d'ontologies del domini immobiliari i de recomanació de serveis per estructurar adequadament els conceptes i relacions del nostre sistema.

1.4 Objectius del sistema

Els objectius principals que ha d'assolir el nostre sistema són:

1.4.1 Objectius funcionals

1. **Classificar solůlicitants automàticament**: A partir de les característiques bàsiques del solůlicitant (edat, nombre de persones, fills, situació laboral, etc.), el sistema ha d'inferir el seu perfil (persona gran, família amb fills, estudiants, parella jove, etc.) sense requerir que el propi usuari s'auto-classifiqui.

2. **Inferir necessitats implícites**: El sistema ha de deduir requeriments que el solůlicitant potser no ha expressat explícitament. Per exemple, si el solůlicitant té fills petits, el sistema ha d'entendre que necessitarà escoles properes encara que no ho hagi demanat directament.

3. **Descartar ofertes inadequades**: Abans de puntuar, el sistema ha d'aplicar filtres durs per eliminar ofertes que clarament no són adequades (fora de pressupost estricte, no permeten mascotes quan és imprescindible, no són accessibles quan cal, etc.).

4. **Puntuar ofertes segons adequació**: Les ofertes que superen els filtres han de rebre una puntuació que reflecteixi el seu grau d'adequació global, considerant tant aspectes de l'habitatge com de l'entorn.

5. **Classificar ofertes en graus de recomanació**: Basant-se en la puntuació, assignar cada oferta a una categoria: parcialment adequada, adequada o molt recomanable.

6. **Explicar les recomanacions**: Per a cada oferta recomanada, el sistema ha de generar explicacions que indiquin per què és adequada (punts forts) i, en cas d'ofertes parcialment adequades, quins criteris no compleix plenament.

1.4.2 Objectius de qualitat

1. **Transparència**: Les decisions del sistema han de ser comprensibles i justificables. L'usuari ha de poder entendre per què una oferta està recomanada i una altra no.

2. **Equitat**: El sistema no ha de discriminar de manera injustificada cap perfil de solůlicitant. Les regles han de reflectir preferències raonables, no biaixos arbitraris.

3. **Cobertura**: El sistema ha de ser capaç de gestionar una àmplia varietat de perfils i ofertes, no només casos ideals o trivials.

4. **Consistència**: Aplicat a situacions similars, el sistema ha de produir recomanacions coherents.

1.5 Resultats del sistema

El sistema proporciona com a sortida una llista de recomanacions personalitzades per a cada solůlicitant, amb la següent informació:

1.5.1 Per a cada solůlicitant

- **Top 3 d'ofertes recomanades**: El sistema presenta les tres millors ofertes ordenades per puntuació, facilitant la presa de decisió sense saturar l'usuari amb massa opcions.

- **Grau de recomanació**: Per a cada oferta, s'indica si és "Parcialment adequada", "Adequada"o "Molt recomanable".

- **Puntuació numèrica**: Tot i que l'usuari final veu principalment la classificació qualitativa, el sistema genera internament una puntuació numèrica que permet ordenar les ofertes amb precisió.

1.5.2 Per a cada oferta recomanada

- **Característiques bàsiques**: Tipus d'habitatge, superfície, nombre de dormitoris i banys, preu mensual, adreça i districte.

- **Punts forts**: Llista de característiques positives que fan l'oferta especialment adequada per al solůlicitant concret. Per exemple: "Té terrassa o balcó (+10p)", "Molt assolellat (+20p)", "Proximitat a escoles (inferida) (+20p)".

- **Aspectes a considerar**: Per a ofertes parcialment adequades o adequades, s'indiquen els criteris que no es compleixen completament. Per exemple: "Preu lleugerament superior al pressupost màxim (Moderat)", "Planta alta sense ascensor (Lleu)".

1.5.3 Informació complementària

El sistema també genera informació interna útil per a depuració i anàlisi:

- **Ofertes descartades**: Registre de quines ofertes s'han descartat i per quin motiu per a cada solůlicitant.
- **Requisits inferits**: Documentació de quines necessitats s'han deduït automàticament per a cada perfil.
- **Proximitats calculades**: Taula de distàncies entre cada habitatge i cada servei, classificades en molt a prop, distància mitjana o lluny.

Aquest disseny de sortida equilibra la utilitat per a l'usuari final (informació clara i accionable) amb la necessitat de transparència i explicabilitat que són fonamentals en un sistema basat en coneixement.

2 Conceptualització

2.1 Conceptes del domini

Per poder resoldre el problema de recomanació d'habitatges de manera efectiva, primer hem hagut d'identificar i organitzar tots els conceptes rellevants del domini. Aquesta tasca no és trivial, ja que el món immobiliari és complex i hi intervenen molts actors i característiques diferents.

2.1.1 Jerarquia de solůlicitants

Un dels conceptes centrals del nostre sistema és el **Solůlicitant**, que representa la persona o grup de persones que busca habitatge. No tots els solůlicitants tenen les mateixes necessitats, i aquesta diferència és fonamental per fer bones recomanacions. Hem identificat una jerarquia basada principalment en l'edat i la situació familiar:

Persones Grans (> 65 anys): Aquest colůlectiu té necessitats molt específiques relacionades amb l'accessibilitat i la proximitat a serveis essencials. Una persona de 70 anys no pot caminar llargues distàncies fins al supermercat ni pujar escales fins a un tercer pis sense ascensor. A més, la proximitat a centres de salut esdevé un factor crític.

Joves (< 35 anys): Dins d'aquest grup hem diferenciat entre:

- *Grups d'Estudiants*: Generalment amb pressupostos ajustats, valoren especialment la proximitat al transport públic i zones d'oci. Necessiten habitatges amb habitacions individuals per mantenir certa privacitat, i prefereixen pisos ja moblats perquè no disposen de mobles propis.
- *Parelles Joves*: Similar als estudiants en algunes preferències (oci, transport), però amb més estabilitat econòmica i interessos diferents (poden estar pensant en tenir fills aviat).
- *Individus Joves*: Persones soles que busquen independència, sovint estudis o pisos petits amb bones comunicacions.

Adults (35-65 anys): Aquest és el grup més heterogeni, i l'hem subdividit segons la seva situació familiar:

- *Parelles amb Fills*: Tenen necessitats clares relacionades amb l'educació dels fills (proximitat a escoles), espais exteriors (parcs, zones verdes) i més habitacions. També valoren la tranquilůlitat del barri.
- *Parelles que Planegen Tenir Fills*: Similar a l'anterior però amb menor urgència. Busquen zones adequades per quan arribin els fills, amb bones escoles properes encara que de moment no les necessitin.
- *Parelles sense Fills*: Més flexibles en ubicació però valoren confort, serveis culturals (teatres, cinemes) i probablement més espai que una parella jove.

- *Individus amb Fills (Famílies Monoparentals)*: Necessitats similars a les parelles amb fills però sovint amb pressupostos més ajustats.
- *Individus sense Fills*: Adults independents amb prioritats professionals i personals variades.

Compradors de Segona Residència: Un cas especial que hem inclòs. Aquestes persones no busquen residència habitual sinó una propietat per vacances o inversió. Valoren especialment vistes, tranquilůlitat, eficiència energètica i qualitats de luxe. No els preocupa tant la proximitat a escoles o serveis quotidians.

2.1.2 Habitatges i les seves característiques

El concepte d'**Habitatge** agrupa totes les propietats físiques i característiques d'un immoble. Hem identificat diversos tipus segons la seva estructura:

Tipus d'habitatge:

- *Pis*: El tipus més comú, en edifici plurifamiliar.
- *Àtic*: Pis a l'última planta, sovint amb terrassa gran.
- *Dúplex*: Habitatge en dues plantes dins d'un edifici.
- *Estudi*: Habitatge molt petit, generalment d'una sola estança.
- *Habitatge Unifamiliar*: Casa independent, ideal per famílies grans.

Característiques físiques bàsiques:

- Superfície habitable (mš)
- Nombre de dormitoris (distingint entre dobles i simples)
- Nombre de banys
- Planta (important per accessibilitat)
- Any de construcció i estat de conservació

Equipament i comoditats:

- Si té terrassa o balcó (i la seva superfície)
- Si està moblat i/o amb electrodomèstics
- Si disposa d'ascensor
- Sistemes de climatització (calefacció, aire condicionat)
- Si permet mascotes
- Si té plaça d'aparcament

- Armaris encastats i traster

Característiques ambientals:

- Orientació solar (matí, tarda, tot el dia, mai)
- Si és exterior o interior
- Nivell de soroll (baix, mitjà, alt)
- Tipus de vistes (mar, muntanya, ciutat, cap)
- Consum energètic (A a G)

2.1.3 Serveis urbans

Els **Serveis** representen tots els equipaments i zones d'interès que hi ha a la ciutat i que poden influir en l'adequació d'un habitatge segons la seva proximitat. Hem creat una taxonomia força completa:

Serveis Educatius:

- Llars d'infants (0-3 anys)
- Escoles (primària i secundària)
- Instituts
- Universitats

Serveis de Salut:

- Centres de salut (CAPs)
- Hospitals
- Farmàcies

Serveis Comercials:

- Supermercats
- Mercats municipals
- Centres comercials
- Hipermercats

Transport:

- Estacions de metro

- Parades de bus
- Estacions de tren
- Autopistes (accés)
- Aeroport

Zones Verdes:

- Parcs
- Jardins
- Zones esportives

Serveis d'Oci:

- Bars i restaurants
- Cinemes i teatres
- Discoteques
- Gimnasos
- Estadis

2.1.4 Localització

La **Localització** és el concepte que fa de pont entre habitatges i serveis. Cada habitatge i cada servei tenen una localització que inclou:

- Coordenades (X, Y) per calcular distàncies
- Adreça completa
- Districte i barri
- Codi postal

2.1.5 Oferta

Una **Oferta** representa un habitatge disponible per llogar en un moment donat. Inclou:

- Referència a l'habitatge concret
- Preu mensual de lloguer
- Data de publicació
- Disponibilitat (si/no)

2.1.6 Conceptes auxiliars

A més dels conceptes principals, hem definit conceptes auxiliars que s'utilitzen durant el procés de raonament:

Proximitat: Relació entre un habitatge i un servei que indica:

- La distància en metres
- Classificació qualitativa (molt a prop, distància mitjana, lluny)
- Categoria del servei

Requisit Inferit: Necessitat que el sistema dedueix automàticament per a un solůlicitant:

- Categoria de servei necessari
- Si és obligatori o preferible
- Motiu de la inferència

Recomanació: Resultat de l'avaluació d'una oferta per a un solůlicitant:

- Puntuació numèrica
- Grau de recomanació (parcialment adequat, adequat, molt recomanable)
- Punts positius
- Criteris no complerts

2.2 Descomposició en problemes i subproblemes

El problema global de recomanar habitatges és massa complex per resoldre'l d'un sol cop. L'hem descomposat en una seqüència de subproblemes que es resolen de manera ordenada, cadascun construint sobre els resultats dels anteriors. Aquesta descomposició segueix la metodologia de classificació heurística, que és especialment adequada per a problemes on hem de categoritzar elements (ofertes) segons múltiples criteris.

2.2.1 Fase 1: Inicialització

Objectiu: Preparar totes les dades base necessàries per al raonament posterior.

Subproblema 1.1: Càrrega de dades

- Carregar la ontologia amb totes les classes i relacions
- Instanciar tots els habitatges, serveis i ofertes disponibles
- Crear o capturar el perfil del solůlicitant

Subproblema 1.2: Càlcul de proximitats

Aquest és un subproblema crucial que prepara informació espacial que s'utilitzarà constantment més endavant. Per a cada parella (habitatge, servei):

1. Obtenir les coordenades de l'habitatge i del servei
2. Calcular la distància de Manhattan: $d = |x_1 - x_2| + |y_1 - y_2|$
3. Classificar aquesta distància:
    - Molt a prop: < 200m
    - Distància mitjana: 200-400m
    - Lluny: > 400m
4. Emmagatzemar aquesta informació com a fet

Per què calculem això per endavant? Perquè quasi totes les regles posteriors necessitaran saber si un habitatge està a prop d'un cert tipus de servei, i és molt més eficient calcular-ho una vegada que repetir el càlcul cada vegada.

Subproblema 1.3: Expansió de categories

Un detall important: també creem relacions de proximitat per a categories generals. Per exemple, si detectem que un habitatge està a prop d'una "Escola", també afirmem que està a prop d'un "Servei Educatiu". Això ens permet escriure regles més generals sense haver de contemplar tots els subtipus específics.

Subproblema 1.4: Crear recomanacions inicials

Per a cada parella (solůlicitant, oferta disponible), creem una recomanació buida amb puntuació 0 i sense grau assignat. Aquestes recomanacions s'aniran omplint en les fases següents.

2.2.2 Fase 2: Abstracció

Objectiu: Transformar les dades brutes del solůlicitant en un perfil significatiu i inferir necessitats implícites.

Subproblema 2.1: Classificació del solůlicitant

A partir de les característiques demogràfiques (edat, nombre de persones, fills, situació laboral), el sistema ha de classificar automàticament el solůlicitant en una de les categories de la jerarquia. Això no és trivial perquè les categories es poden solapar i cal establir prioritats:

1. Si és segona residència $\rightarrow$ CompradorSegonaResidència
2. Altrament, si edat > 65 $\rightarrow$ PersonaGran
3. Altrament, si estudia a la ciutat $\rightarrow$ GrupEstudiants
4. Altrament, si edat ≤ 35 i més d'1 persona $\rightarrow$ ParellaJove

5. Altrament, si edat ≤ 35 i 1 persona Jove
6. Altrament, si edat > 35:
    - Si té fills ParellaAmbFills o IndividuAmbFills (segons núm. persones)
    - Si planeja fills ParellaFutursFills o IndividuFutursFills
    - Altrament ParellaSenseFills o IndividuSenseFills

Aquesta classificació és determinista però té en compte múltiples factors i segueix un ordre de prioritat basat en el nostre coneixement del domini.

Subproblema 2.2: Inferència de requisits

Un cop classificat el solůlicitant, podem inferir necessitats que probablement no haurà expressat explícitament. Aquest coneixement prové de l'experiència experta en el sector immobiliari:

Per a Famílies amb Fills:

- Necessita serveis educatius propers (escoles)
- Prefereix zones verdes (parcs on els nens puguin jugar)

Per a Persones Grans:

- Necessita serveis de salut propers
- Necessita serveis comercials propers (menys mobilitat)

Per a Estudiants:

- Necessita transport públic
- Prefereix zones d'oci

Per a Parelles amb Plans de Fills:

- Prefereix escoles properes (per al futur)
- Prefereix zones verdes

Per a Joves en edat laboral sense vehicle:

- Necessita transport públic

Per a persones amb vehicle que treballen fora de la ciutat:

- Prefereix proximitat a autopistes

Aquests requisits s'emmagatzemen com a fets, indicant si són obligatoris (han de complir-se sí o sí) o preferibles (sumaran punts però no són imprescindibles). De moment, hem definit la majoria com a preferibles per evitar descartar massa ofertes.

2.2.3 Fase 3: Descart

Objectiu: Eliminar ofertes que clarament no són adequades abans de fer cap càlcul de puntuació.

Aquest subproblema implementa restriccions dures que no admeten excepcions. Si una oferta viola alguna d'aquestes restriccions, es descarta directament i ja no es considera més:

Subproblema 3.1: Restriccions econòmiques

- Si el preu supera el pressupost màxim i el solůlicitant té marge estricte DESCARTAR
- Si el preu és inferior al pressupost mínim i el solůlicitant té marge estricte DESCARTAR

El pressupost mínim pot semblar estrany, però alguns solůlicitants el defineixen perquè consideren que ofertes massa barates poden amagar problemes o ser estafes.

Subproblema 3.2: Restriccions sobre mascotes

- Si el solůlicitant té mascotes i l'habitatge no les permet DESCARTAR

Subproblema 3.3: Restriccions d'accessibilitat

- Si el solůlicitant necessita accessibilitat i l'habitatge està en planta alta sense ascensor DESCARTAR

Subproblema 3.4: Serveis a evitar

- Si el solůlicitant vol evitar un cert tipus de servei (ex: discoteques) i n'hi ha un molt a prop DESCARTAR

Subproblema 3.5: Requisits inferits obligatoris

- Si s'ha inferit un requisit com a obligatori i no hi ha cap servei d'aquesta categoria a prop o distància mitjana DESCARTAR

De moment aquesta darrera regla no s'activa gaire perquè hem marcat la majoria de requisits com a preferibles, però és important tenir-la per a casos més extrems.

Subproblema 3.6: Superfície mínima

- Si la superfície és inferior a 10mš per persona DESCARTAR

Aquest és un criteri de sentit comú: 3 persones no poden viure dignament en 25mš.

Subproblema 3.7: Restriccions específiques de perfil

Alguns perfils tenen restriccions addicionals:

- Estudiants: descartar habitatges a reformar (no tenen temps ni diners)
- Persones amb vehicle: descartar habitatges sense aparcament
- Persones grans: descartar si serveis de salut estan lluny
- Compradors segona residència: descartar estudis (massa petits)

2.2.4 Fase 4: Puntuació (Scoring)

Objectiu: Assignar punts a les ofertes que han superat el descart segons la seva adequació.

Aquest és el subproblema més complex perquè és on rau la major part del coneixement expert. Hem dissenyat un sistema de puntuació additiu on cada característica positiva suma punts i cada deficiència en resta. L'estratègia és:

Subproblema 4.1: Puntuació per adequació de pressupost

El preu és fonamental, però no és només "dins pressupost"o "fora pressupost":

- Preu excepcional (>30% d'estalvi sobre el màxim): +50 punts
- Preu molt bo (20-30% d'estalvi): +40 punts
- Preu perfecte (dins el rang mínim-màxim): +30 punts
- Preu adequat (marge flexible, fins 15% sobre màxim): +20 punts
- Preu lleugerament alt (dins marge flexible): -10 punts

Subproblema 4.2: Puntuació per característiques de l'habitatge

Diferents perfils valoren diferents coses:

Característiques generals (sumen per a tots):

- Té terrassa o balcó: +20 punts
- Molt assolellat (tot el dia): +20 punts
- Té vistes: +20 punts
- Té piscina comunitària: +20 punts

Per a Joves:

- Terrassa (extra): +5 punts (els joves valoren molt les terrasses)
- Piscina (extra): +5 punts
- Amb electrodomèstics: +20 punts

Per a Estudiants:

- Moblat: +20 punts (imprescindible per a estudiants)

Per a Adults i Persones Grans:

- Aire condicionat: +20 punts
- Calefacció: +20 punts
- Traster: +20 punts

Per a Persones Grans i Segones Residències:

- Nivell de soroll baix: +20 punts

Per a Segones Residències:

- Vistes al mar o muntanya: +20 punts
- Consum energètic A o B: +20 punts
- A reformar: +20 punts (poden fer-la a mida amb pressupost)

Per a Famílies i Estudiants:

- Més d'un bany: +20 punts
- Piscina: +5 punts

Per a Parelles i Famílies:

- Almenys una habitació doble: +20 punts

Per a Estudiants i Individus:

- Suficients habitacions individuals: +20 punts

Subproblema 4.3: Puntuació per serveis propers

Aquí és on s'apliquen els requisits inferits i les preferències:
Transport públic (molt a prop o distància mitjana):

- Per a Joves i Estudiants: +10 punts

Universitat (molt a prop o distància mitjana):

- Per a Joves: +20 punts

Autopista (molt a prop o distància mitjana):

- Si s'ha inferit la necessitat: +20 punts

Serveis comercials (molt a prop o distància mitjana):

- Per a Adults: +20 punts

Serveis de salut:

- Molt a prop per a Persones Grans: +10 punts addicionals

Oci:

- Transport, bars, discoteques, gimnasos per a Joves: +10 punts
- Cinemes, teatres, restaurants per a Adults: +10 punts
- Teatres, restaurants per a Persones Grans: +10 punts

Servei preferit explícitament:

- Si el solůlicitant ha indicat que prefereix un servei i està molt a prop o distància mitjana: +5 punts

Requisit inferit satisfet:

- Si un requisit inferit (escola per fills, zona verda, etc.) està molt a prop: +20 punts

Subproblema 4.4: Penalitzacions

També restem punts per deficiències:

- Nivell de soroll alt: -10 punts
- Planta alta sense ascensor: -10 punts
- Poca llum natural (orientació "Mai"): -10 punts
- Baixa eficiència energètica (F o G): -10 punts

Subproblema 4.5: Registre de punts positius i criteris no complerts

Mentrestant que anem puntuant, també generem fets que expliquen per què s'han donat o restat punts:

- `punt-positiu`: "Té terrassa o balcó", "Pressupost perfecte", etc.
- `criteri-no-complert`: "Nivell de soroll alt", "Planta alta sense ascensor", etc.

Aquests fets són els que després es mostraran a l'usuari per justificar la recomanació.

2.2.5 Fase 5: Classificació

Objectiu: Assignar un grau de recomanació qualitatiu a cada oferta segons la seva puntuació.

Aquest és un subproblema relativament simple un cop tenim les puntuacions. Apliquem uns llindars:

- Puntuació ≥ 70: **Molt Recomanable**
- Puntuació ≥ 40: **Adequat**
- Puntuació > 0: **Parcialment Adequat**
- Puntuació ≤ 0: No es recomana (no s'inclou en els resultats)

Aquests llindars s'han escollit experimentalment per assegurar que les categories tinguin sentit. Una oferta "Molt Recomanable"ha de tenir diversos punts forts, no només complir el mínim.

2.2.6 Fase 6: Presentació

Objectiu: Generar la sortida en un format útil per a l'usuari.

Subproblema 6.1: Ordenació

Per a cada solůlicitant, ordenem les seves recomanacions per puntuació descendent.

Subproblema 6.2: Selecció del Top 3

Prenem només les 3 millors ofertes per solůlicitant per no saturar l'usuari amb massa informació.

Subproblema 6.3: Formatació i explicació

Per a cada oferta del Top 3, mostrem:

- Dades bàsiques de l'habitatge
- Grau de recomanació
- Llista de punts forts (extrets dels fets `punt-positiu`)
- Llista d'aspectes a considerar (extrets dels fets `criteri-no-complert`)

2.3 Exemples de coneixement expert

Per ilůlustrar millor com hem capturat el coneixement expert del domini, presentem alguns exemples concrets de regles i el raonament que hi ha darrere:

2.3.1 Exemple 1: Inferència per a famílies amb fills

Coneixement expert: L·Les famílies amb fills petits necessiten tenir escoles properes i zones verdes on els nens puguin jugar. Això no és negociable, encara que la família no ho demani explícitament."

Formalització:

```
SI solůlicitant.numeroFills > 0
LLAVORS
    - Inferir necessitat de ServeiEducatiu (preferible)
    - Inferir necessitat de ZonaVerda (preferible)
    - Motiu: "Família amb fills necessita escoles i zones verdes"
```

Impacte: Quan es puntuen ofertes per aquesta família, les que tinguin escoles i parcs molt a prop rebran +20 punts extres. Les que no en tinguin no es descartaran (perquè no és obligatori), però quedaran pitjor posicionades.

2.3.2 Exemple 2: Descart per accessibilitat

Coneixement expert: Üna persona que necessita accessibilitat (sigui per edat avançada, per problemes de mobilitat o per conviure amb persones grans) no pot viure en un pis alt sense ascensor. Això és una barrera infranquejable."

Formalització:

```
SI solůlicitant.necessitaAccessibilitat = si
   I habitatge.teAscensor = no
   I habitatge.plantaPis > 0
LLAVORS
    DESCARTAR oferta
    Motiu: "No accessible: sense ascensor i planta alta"
```

Impacte: Aquestes ofertes mai apareixeran en les recomanacions per a solůlicitants que necessitin accessibilitat, independentment de quantes altres virtuts tinguin.

2.3.3 Exemple 3: Valoració diferencial del preu

Coneixement expert: "Trobar un habitatge un 30% més barat del que estaves disposat a pagar és una oportunitat excepcional que cal destacar. Però un habitatge lleugerament més car del màxim (10-15%) pot ser acceptable si té altres virtuts, sempre que el client no hagi dit que el pressupost és infranquejable."

Formalització:

```
SI preu < pressupostMaxim * 0.7
LLAVORS punts += 50, motiu: "Preu excepcional (>30% estalvi)"

SI pressupostMinim <= preu <= pressupostMaxim
```

```
LLAVORS punts += 30, motiu: "Pressupost perfecte"

SI pressupostMaxim < preu <= pressupostMaxim * 1.15
   I margeEstricte = no
LLAVORS
    punts += 20, motiu: "Pressupost adequat"
    PERO punts -= 10 addicionals
    I registrar: criteri-no-complert "Preu lleugerament superior"
```

Impacte: Un habitatge perfecte però 10% més car pot ser "Molt Recomanable"si té molts punts forts, però s'advertirà l'usuari que supera el pressupost. Un habitatge 30% més barat sumarà molts punts i potser compensarà algunes mancances.

2.3.4 Exemple 4: Estudiants i habitatges a reformar

Coneixement expert: "Els estudiants necessiten un habitatge on puguin entrar a viure immediatament. No tenen temps, diners ni coneixements per reformar un pis. Per tant, descartem directament qualsevol habitatge que estigui catalogat com 'a reformar'."

Formalització:

```
SI solůlicitant ES-UN GrupEstudiants
   I habitatge.estatConservacio = "AReformar"
LLAVORS
    DESCARTAR oferta
    Motiu: "Estudiants necessiten habitatge llest per habitar"
```

Contrapunt: Per a compradors de segona residència, un habitatge a reformar suma +20 punts, perquè ells sí tenen recursos i volen personalitzar-lo.

2.3.5 Exemple 5: Persones grans i serveis de salut

Coneixement expert: "Per a una persona gran, tenir un centre de salut molt a prop és molt més valuós que tenir-lo a distància mitjana, perquè la mobilitat és limitada. A més, si els serveis de salut estan lluny, l'habitatge directament no és adequat."

Formalització:

```
// Inferència
SI solůlicitant ES-UN PersonaGran
LLAVORS
    Inferir necessitat de ServeiSalut (preferible)

// Descart
SI solůlicitant ES-UN PersonaGran
   I NO existeix ServeiSalut a MoltAProp o DistanciaMitjana
LLAVORS
    DESCARTAR oferta
```

```
    Motiu: "Serveis de salut massa lluny per persona gran"

// Puntuació extra per molt a prop
SI solůlicitant ES-UN PersonaGran
   I existeix ServeiSalut a MoltAProp
LLAVORS
    punts += 10 addicionals
    Motiu: "Centre de salut molt proper (ideal per persona gran)"
```

Impacte: Les persones grans només veuran ofertes amb centres de salut relativament propers, i dins d'aquestes, es prioritzaran les que els tinguin molt a prop.

2.3.6 Exemple 6: Compradors de segona residència

Coneixement expert: "Qui compra una segona residència busca una cosa diferent: no li preocupen escoles ni supermercats propers, però sí les vistes, la tranquilůlitat, la qualitat de l'habitatge i l'eficiència energètica (perquè la casa pot estar buida molts dies). També poden permetre's comprar quelcom a reformar per fer-ho al seu gust."

Formalització:

```
SI solůlicitant ES-UN CompradorSegonaResidencia
LLAVORS:
    // Valoren vistes especials
    SI habitatge.tipusVistes = "Mar" O "Muntanya"
    LLAVORS punts += 20

    // Valoren tranquilůlitat
    SI habitatge.nivellSoroll = "Baix"
    LLAVORS punts += 20

    // Valoren eficiència
    SI habitatge.consumEnergetic = "A" O "B"
    LLAVORS punts += 20

    // No els molesta reformar
    SI habitatge.estatConservacio = "AReformar"
    LLAVORS punts += 20

    // Descarten estudis (massa petits)
    SI habitatge ES-UN Estudi
    LLAVORS DESCARTAR
```

2.4 Descripció informal del procés de resolució

Ara que hem vist tots els components per separat, descrivim com funciona el sistema de manera integrada quan un solůlicitant vol trobar habitatge:

2.4.1 Escenari d'ús típic

Imaginem que una família de 4 membres (dos adults de 38 anys i dos fills de 6 i 10 anys) busca pis a Barcelona. Tenen un pressupost màxim de 1.500 mensuals (però són una mica flexibles), tenen un gos, i el pare treballa a la ciutat mentre la mare teletreballa.

Pas 1: Captura de dades

El sistema pregunta o rep les dades bàsiques:

- Nom: "Família Garcia"
- Edat del solůlicitant principal: 38 anys
- Nombre de persones: 4
- Fills: 2 (edats: 6, 10)
- Pressupost: 600 - 1.500 (flexible)
- Té mascotes: Sí (1 gos)
- Té vehicle: Sí
- Necessita accessibilitat: No
- Treballa a la ciutat: Sí

Pas 2: Inicialització

El sistema:

- Carrega totes les ofertes disponibles (diguem-ne 8)
- Calcula distàncies entre els 8 habitatges i tots els serveis de la ciutat (metro, escoles, hospitals, parcs, etc.)
- Crea 8 recomanacions buides (una per oferta)

Pas 3: Abstracció

El sistema analitza el perfil:

- Edat $38 > 35$ és adult
- Té fills és ParellaAmbFills

Després infereix necessitats:

- "Família amb fills necessita escoles" Requisit: ServeiEducatiu (preferible)
- "Família amb fills prefereix zones verdes" Requisit: ZonaVerda (preferible)

Pas 4: Descart

El sistema revisa les 8 ofertes:

- Oferta 1: Pis de 95mš, 1.350, permet mascotes PASSA
- Oferta 2: Àtic de 120mš, 1.800, permet mascotes PASSA (preu alt però flexible)
- Oferta 3: Estudi de 35mš, 650, NO permet mascotes DESCARTAT (no mascotes)
- Oferta 4: Pis de 70mš, 1.100, NO permet mascotes DESCARTAT (no mascotes)
- Oferta 5: Casa de 200mš, 3.500, permet mascotes DESCARTAT (preu excessiu)
- Oferta 6: Pis de 80mš, 950, NO permet mascotes DESCARTAT (no mascotes)
- Oferta 7: Pis de 95mš, 1.650, permet mascotes PASSA (marge flexible)
- Oferta 8: Estudi de 35mš, 200, NO permet mascotes DESCARTAT (massa petit i preu sospitós)

Queden 3 ofertes: 1, 2 i 7.

Pas 5: Puntuació

Per a l'Oferta 1 (pis de 95mš, 1.350):

- Preu dins pressupost perfecte: +30 punts
- Té terrassa: +20 punts
- Assolellat tot el dia: +20 punts
- Té vistes: +20 punts
- Té plaça d'aparcament: inherent (no suma extra perquè és necessari)
- Escola molt a prop (requisit inferit satisfet): +20 punts
- Parc a distància mitjana (requisit inferit satisfet): +20 punts
- 2 habitacions dobles: +20 punts
- Més d'un bany: +20 punts
- Total: 170 punts

Per a l'Oferta 2 (àtic de 120mš, 1.800):

- Preu lleugerament superior (1.800 vs 1.500): +20 - 10 = +10 punts net
- Registra: criteri-no-complert "Preu 20% superior"
- Terrassa gran (40mš): +20 punts
- Assolellat tot el dia: +20 punts
- Vistes a la muntanya: +20 punts
- Escola a distància mitjana: +20 punts

- Parc a distància mitjana: +20 punts
- 2 habitacions dobles: +20 punts
- 2 banys: +20 punts
- Àtic (planta 5): +0 (no hi ha regla específica, però és un plus implícit)
- Total: 150 punts

Per a l'Oferta 7 (pis de 95mš, 1.650):

- Preu superior però dins marge: +20 - 10 = +10 punts
- Registra: criteri-no-complert "Preu 10% superior"
- Sense terrassa: 0
- Orientació matí (no tot el dia): 0
- Sense vistes destacables: 0
- NO té escola propera: 0 (i no suma el bonu)
- Parc lluny: 0
- 1 habitació doble: +20 punts
- 1 bany: 0
- Total: 30 punts

Pas 6: Classificació

Segons les puntuacions:

- Oferta 1: 170 punts **Molt Recomanable**
- Oferta 2: 150 punts **Molt Recomanable**
- Oferta 7: 30 punts **Parcialment Adequat**

Pas 7: Presentació

El sistema mostra el Top 3 (que en aquest cas són les 3 úniques que queden):

```
SOLůLICITANT: Família Garcia

#1 - oferta-1 - *** Molt Recomanable *** (170 punts)
    Tipus: Pis
    Superfície: 95 mš
    Dormitoris: 3 | Banys: 2
    Preu: 1.350 EUR/mes
    Adreça: Carrer Aragó 250
```

```
    Districte: Eixample
    ----------------------------------------
    PUNTS FORTS:
      [+] Pressupost perfecte (+30p)
      [+] Té terrassa o balcó (+20p)
      [+] Molt assolellat (+20p)
      [+] Té bones vistes (+20p)
      [+] Preferència detectada molt a prop: Escola (+20p)
      [+] Preferència detectada molt a prop: Parc (+20p)
      [+] Disposa d'habitació doble (+20p)
      [+] Més d'un bany (+20p)

#2 - oferta-2 - *** Molt Recomanable *** (150 punts)
    Tipus: Àtic
    Superfície: 120 mš
    Dormitoris: 3 | Banys: 2
    Preu: 1.800 EUR/mes
    Adreça: Carrer Verdi 45
    Districte: Gràcia
    ----------------------------------------
    PUNTS FORTS:
      [+] Té terrassa o balcó (+20p)
      [+] Molt assolellat (+20p)
      [+] Té bones vistes (+20p)
      [+] Disposa d'habitació doble (+20p)
      [+] Més d'un bany (+20p)
    ASPECTES A CONSIDERAR:
      Preu lleugerament superior al pressupost (Moderat)

#3 - oferta-7 - *** Parcialment Adequat *** (30 punts)
    Tipus: Pis
    Superfície: 95 mš
    Dormitoris: 3 | Banys: 1
    Preu: 1.650 EUR/mes
    Adreça: Carrer València 180
    Districte: Eixample
    ----------------------------------------
    PUNTS FORTS:
      [+] Disposa d'habitació doble (+20p)
    ASPECTES A CONSIDERAR:
      Preu lleugerament superior al pressupost (Moderat)
      No hi ha escoles molt properes
      Zones verdes lluny
```

2.4.2 Flux general del sistema

En resum, el procés segueix sempre aquesta seqüència:

1. **Entrada** Dades del solůlicitant
2. **Inicialització** Càlcul de proximitats, creació de recomanacions buides
3. **Abstracció** Classificació del solůlicitant, inferència de necessitats
4. **Descart** Eliminació d'ofertes inadequades (filtre dur)
5. **Puntuació** Avaluació detallada de les ofertes restants
6. **Classificació** Assignació de grau de recomanació
7. **Presentació** Generació del Top 3 amb explicacions
8. **Sortida** Recomanacions personalitzades

Aquesta descomposició en fases amb objectius clars fa que el sistema sigui modular, mantenible i fàcil d'entendre. Cada fase prepara les dades per a la següent, i el coneixement expert es distribueix de manera natural entre les diferents regles de cada fase.

3 Formalització

3.1 Disseny de l'Ontologia

L'ontologia s'ha implementat utilitzant Protégé i exportada a CLIPS. A continuació es detalla l'estructura formal de classes i propietats.

3.1.1 Jerarquia de Classes

L'arbre de classes principal és el següent:

```
USER
+-- Solicitant
|   +-- Joves (GrupEstudiants, ParellaJove)
|   +-- Adults (ParellaAmbFills, Individu, etc.)
|   +-- PersonaGran
|   +-- CompradorSegonaResidencia
+-- Habitatge
|   +-- Pis
|   +-- Atic
|   +-- Duplex
|   +-- HabitatgeUnifamiliar
+-- Servei
|   +-- ServeiSalut (Hospital, CAP...)
|   +-- ServeiOci (Bar, Cinema...)
|   +-- Transport (Metro, Bus...)
+-- Oferta
+-- Localitzacio
```

3.1.2 Atributs i Relacions (Slots)

Els atributs s'han definit amb tipus estrictes per facilitar el raonament a les regles:

- **Relacions d'Objectes:**
  - `teLocalitzacio`: Vincula Habitatge/Servei amb Localitzacio.
  - `teHabitatge`: Vincula Oferta amb Habitatge.
  - `prefereixServei` / `evitaServei`: Multislots a Solicitant que contenen instàncies de serveis específics.
- **Atributs de Dades:**
  - `preuMensual` (FLOAT): Per comparacions numèriques.
  - `disponible` (SYMBOL: si/no).
  - `orientacioSolar` (STRING: "TotElDia", "Mati", etc.).

3.2 Model de Raonament i Regles

El sistema utilitza un motor d'inferència basat en regles de producció (forward chaining) organitzat en mòduls o fases temporals.

3.2.1 Fase 1: Abstracció i Càlcul Espacial

Es defineix una funció de distància euclidiana:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Les regles categoritzen aquesta distància en conceptes qualitatius:

- $d < 500m \rightarrow$ `MoltAProp`
- $500m \leq d < 1000m \rightarrow$ `DistanciaMitjana`
- $d \geq 1000m \rightarrow$ `Lluny`

Això permet escriure regles com: "Si hi ha una escola `MoltAProp`, suma punts".

3.2.2 Fase 2: Inferència de Requisits

En aquesta fase es generen fets intermedis `requisit-inferit`.

- *Regla:* Si `numeroFills > 0` → Assert `requisit-inferit(Educacio, Obligatori: no, Motiu: "Família")`.
- *Regla:* Si `classe és PersonaGran` → Assert `requisit-inferit(Salut, Obligatori: si)`.

3.2.3 Fase 3: Descart (Hard Constraints)

S'eliminen les ofertes que violen restriccions crítiques. Es genera un fet `oferta-descartada` i es documenta el motiu. Exemples de regles de descart:

- Preu > Pressupost Màxim (si el marge és estricte).
- Habitatge sense ascensor AND Pis > 0 AND Solůlicitant requereix accessibilitat.
- Habitatge sense mobles AND Solůlicitant és GrupEstudiants.

3.2.4 Fase 4: Scoring (Soft Constraints)

Les ofertes supervivents reben punts. Es parteix d'una puntuació base (0) i es modifica mitjançant `modify ?rec`.

- +20 punts si té habitació doble (per parelles).

- +25 punts si el transport públic és `MoltAProp`.
- -10 punts si el preu supera el pressupost però està dins del marge flexible (15%).
- +50 punts si és una "ganga"(preu $<$ 70% del pressupost).

4 Implementació

4.1 Representació de l'ontologia

Un dels primers reptes que ens vam trobar va ser com traduir l'ontologia que havíem dissenyat en Protégé a estructures de dades utilitzables en CLIPS. CLIPS ofereix el paradigma COOL (CLIPS Object-Oriented Language) que ens permet representar jerarquies de classes i instàncies d'una manera molt natural.

4.1.1 Traducció automàtica amb owl2clips

Protégé permet exportar ontologies en format OWL, i existeix una eina anomenada `owl2clips` que tradueix automàticament aquests fitxers a codi CLIPS. Aquesta eina ens ha estalviat molt de treball manual i ha garantit que la representació en CLIPS sigui fidel a l'ontologia original.

El procés que vam seguir va ser:

1. Dissenyar i documentar l'ontologia en Protégé
2. Exportar-la com `ontologia.owl`
3. Utilitzar `owl2clips` per generar `ontologia.clp`
4. Revisar i ajustar el codi generat si calia

4.1.2 Estructura de les classes

La traducció genera `defclass` per a cada concepte de l'ontologia. Per exemple, la jerarquia de solůlicitants es representa així:

```
(defclass Solicitant
    (is-a USER)
    (role concrete)
    (pattern-match reactive)
    (slot nom (type STRING))
    (slot edat (type INTEGER))
    (slot numeroPersones (type INTEGER))
    (slot numeroFills (type INTEGER))
    (multislot edatsFills (type INTEGER))
    (slot pressupostMaxim (type FLOAT))
    (slot pressupostMinim (type FLOAT))
    (slot margeEstricte (type SYMBOL))
    (slot teMascotes (type SYMBOL))
    (slot numeroMascotes (type INTEGER))
    (slot tipusMascota (type STRING))
    ; ... més atributs
)

(defclass PersonaGran
```

```
    (is-a Solicitant)
    (role concrete)
    (pattern-match reactive)
)

(defclass Joves
    (is-a Solicitant)
    (role concrete)
    (pattern-match reactive)
)

(defclass GrupEstudiants
    (is-a Joves)
    (role concrete)
    (pattern-match reactive)
)
```

Listing 1: Jerarquia de solicitants en CLIPS

L'herència es defineix amb `is-a`, de manera que `GrupEstudiants` hereta tots els atributs de `Joves`, que al seu torn hereta de `Solicitant`. Això ens permet escriure regles generals per a "qualsevol Jove"que s'aplicaran tant a `GrupEstudiants` com a `ParellaJove`.

4.1.3 Representació d'habitatges i serveis

De manera similar, els habitatges i serveis es representen amb les seves respectives jerarquies:

```
(defclass Habitatge
    (is-a USER)
    (role concrete)
    (pattern-match reactive)
    (slot superficieHabitable (type FLOAT))
    (slot numeroDormitoris (type INTEGER))
    (slot numeroDormitorisDobles (type INTEGER))
    (slot numeroDormitorisSimples (type INTEGER))
    (slot numeroBanys (type INTEGER))
    (slot plantaPis (type INTEGER))
    (slot moblat (type SYMBOL))
    (slot ambElectrodomestics (type SYMBOL))
    (slot teAscensor (type SYMBOL))
    (slot permetMascotes (type SYMBOL))
    (slot teTerrassaOBalco (type SYMBOL))
    (slot superficieTerrassa (type FLOAT))
    (slot orientacioSolar (type STRING))
    (slot nivellSoroll (type STRING))
    (slot teLocalitzacio (type INSTANCE))
    ; ... més atributs
)

(defclass Pis (is-a Habitatge))
```

```
(defclass Atic (is-a Habitatge))
(defclass Estudi (is-a Habitatge))
(defclass HabitatgeUnifamiliar (is-a Habitatge))
(defclass Duplex (is-a Habitatge))
```

Listing 2: Jerarquia d'habitatges

Per als serveis, la jerarquia és més complexa perquè tenim categories i subcategories:

```
(defclass Servei
    (is-a USER)
    (role concrete)
    (slot nomServei (type STRING))
    (slot teLocalitzacio (type INSTANCE))
)

(defclass ServeiEducatiu (is-a Servei))
(defclass Escola (is-a ServeiEducatiu))
(defclass Institut (is-a ServeiEducatiu))
(defclass Universitat (is-a ServeiEducatiu))

(defclass ServeiSalut (is-a Servei))
(defclass Hospital (is-a ServeiSalut))
(defclass CentreSalut (is-a ServeiSalut))
(defclass Farmacia (is-a ServeiSalut))

(defclass Transport (is-a Servei))
(defclass EstacioMetro (is-a Transport))
(defclass ParadaBus (is-a Transport))
(defclass Autopista (is-a Transport))
```

Listing 3: Jerarquia de serveis (fragment)

Aquesta jerarquia ens permet fer queries com "troba'm tots els serveis educatius"sense haver d'enumerar Escola, Institut, Universitat, etc.

4.1.4 Relacions entre objectes

Les relacions entre objectes es representen mitjançant slots de tipus `INSTANCE`. Per exemple, un habitatge té una localització:

```
(slot teLocalitzacio (type INSTANCE))
```

I una oferta té un habitatge:

```
(defclass Oferta
    (is-a USER)
    (slot teHabitatge (type INSTANCE))
    (slot preuMensual (type FLOAT))
    (slot disponible (type SYMBOL))
    (slot dataPublicacio (type STRING))
)
```

Això ens permet navegar per les relacions fàcilment. Per exemple, per obtenir la superfície d'un habitatge a partir d'una oferta:

```
?of <- (object (is-a Oferta) (teHabitatge ?hab))
?h <- (object (is-a Habitatge) (name ?hab)
              (superficieHabitable ?sup))
```

4.1.5 Templates per a fets temporals

A més de l'ontologia principal, hem definit diversos templates per representar informació temporal que es genera durant el raonament:

```
(deftemplate proximitat
    (slot habitatge (type INSTANCE))
    (slot servei (type INSTANCE))
    (slot categoria (type SYMBOL))
    (slot distancia (type SYMBOL))  ; MoltAProp,
       DistanciaMitjana, Lluny
    (slot metres (type FLOAT))
)

(deftemplate requisit-inferit
    (slot solicitant (type INSTANCE))
    (slot categoria-servei (type SYMBOL))
    (slot obligatori (type SYMBOL))
    (slot motiu (type STRING))
)

(deftemplate oferta-descartada
    (slot solicitant (type INSTANCE))
    (slot oferta (type INSTANCE))
    (slot motiu (type STRING))
)

(deftemplate Recomanacio
    (slot solicitant (type INSTANCE))
    (slot oferta (type INSTANCE))
    (slot puntuacio (type INTEGER) (default 0))
    (slot grau (type SYMBOL) (default NULL))
)

(deftemplate punt-positiu
    (slot solicitant (type INSTANCE))
    (slot oferta (type INSTANCE))
    (slot descripcio (type STRING))
    (slot punts (type INTEGER) (default 10))
)

(deftemplate criteri-no-complert
    (slot solicitant (type INSTANCE))
    (slot oferta (type INSTANCE))
```

```
39      (slot criteri (type STRING))
40      (slot gravetat (type SYMBOL) (default Lleu))
41  )
```

Listing 4: Templates auxiliars

Aquests templates són crucials per emmagatzemar els resultats intermedis del raonament i per poder generar explicacions al final.

4.2 Modularització del sistema

Un dels punts forts de la nostra implementació és la clara separació en mòduls que corresponen als subproblemes identificats a la fase de conceptualització. Aquesta modularització no només fa el codi més mantenible, sinó que també facilita el desenvolupament incremental.

4.2.1 Control de flux amb fases

Per implementar la descomposició en fases, hem utilitzat un template senzill:

```
(deftemplate fase
   (slot actual (type SYMBOL))
)

(deffacts inicial
    (fase (actual init))
)
```

Cada fase té regles que s'executen només quan la fase actual coincideix, i al final de cada fase una regla de baixa saliència canvia la fase:

```
(defrule init-fi
    "Marca el final de la fase d'inicialitzacio"
    (declare (salience -10))
    ?f <- (fase (actual init))
    =>
    (modify ?f (actual abstraccio))
    (printout t crlf "=== FASE INICIALITZACIO COMPLETADA
       ===" crlf)
)
```

Aquest mecanisme és simple però efectiu: les regles normals tenen saliència per defecte (0), mentre que les regles de transició tenen saliència negativa (-10), de manera que només s'executen quan ja no queden regles normals per disparar.

4.2.2 Fase d'inicialització

Les regles d'aquesta fase tenen la condició `(fase (actual init))` i s'encarreguen de:

Càlcul de proximitats:

```
(defrule abstraccio-calcular-proximitats
    "Calcula la proximitat entre cada habitatge i cada
       servei"
    (fase (actual init))
    ?hab <- (object (is-a Habitatge) (teLocalitzacio ?locH
       ))
    ?locHab <- (object (is-a Localitzacio) (name ?locH)
                       (coordenadaX ?x1) (coordenadaY ?y1)
                          )
    ?serv <- (object (is-a Servei) (teLocalitzacio ?locS))
    ?locServ <- (object (is-a Localitzacio) (name ?locS)
                        (coordenadaX ?x2) (coordenadaY ?y2
                           ))
    (not (proximitat (habitatge ?hab) (servei ?serv)))
    =>
    (bind ?metres (calcular-distancia ?x1 ?y1 ?x2 ?y2))
    (bind ?dist (classificar-distancia ?metres))
    (bind ?cat (class ?serv))
    (assert (proximitat (habitatge ?hab) (servei ?serv)
                        (categoria ?cat) (distancia ?dist)
                        (metres ?metres)))
)
```

Listing 5: Càlcul de distàncies

Aquesta regla itera sobre totes les combinacions (habitatge, servei) i crea un fet de proximitat per a cada una. La condició `(not (proximitat ...))` assegura que no calculem la mateixa distància dues vegades.

Les funcions auxiliars `calcular-distancia` i `classificar-distancia` encapsulen la lògica:

```
(deffunction calcular-distancia (?x1 ?y1 ?x2 ?y2)
    (sqrt (+ (** (- ?x2 ?x1) 2) (** (- ?y2 ?y1) 2)))
)

(deffunction classificar-distancia (?metres)
    (if (< ?metres 200.0) then MoltAProp
    else (if (< ?metres 400.0) then DistanciaMitjana
    else Lluny))
)
```

Expansió de categories:

Per fer les regles posteriors més generals, també creem proximitats per a categories pare:

```
(defrule abstraccio-expandir-categories
    "Expandeix categories pare per evitar resultats buits"
    (declare (salience 99))
    (fase (actual init))
    (proximitat (habitatge ?h) (servei ?s)
                (categoria ?cat) (distancia ?d) (metres ?m
                   ))
```

```
    =>
    ; Transport
    (if (or (eq ?cat EstacioMetro) (eq ?cat ParadaBus)
            (eq ?cat EstacioTren)) then
        (assert (proximitat (habitatge ?h) (servei ?s)
                (categoria TransportPublic)
                (distancia ?d) (metres ?m))))
    ; Educacio
    (if (or (eq ?cat Escola) (eq ?cat Universitat)) then
        (assert (proximitat (habitatge ?h) (servei ?s)
                (categoria ServeiEducatiu)
                (distancia ?d) (metres ?m))))
    ; ... més categories
)
```

Aquesta regla té saliència alta (99) per executar-se immediatament després de crear cada proximitat específica.

Creació de recomanacions inicials:

```
(defrule crear-recomanacions-inicials
    "Crea tot el parell de recomanacions entre oferta i
       solicitant"
    (fase (actual init))
    ?sol <- (object (is-a Solicitant))
    ?of  <- (object (is-a Oferta) (disponible si))
    (not (Recomanacio (solicitant ?sol) (oferta ?of)))
    =>
    (assert (Recomanacio (solicitant ?sol) (oferta ?of)))
)
```

4.2.3 Fase d'abstracció

Aquesta fase té dues responsabilitats: classificar solůlicitants i inferir necessitats.

Classificació de solůlicitants:

Quan el solůlicitant s'ha creat a partir del mode interactiu, tenim un template `dades-solicitant` amb tota la informació bruta. Les regles d'abstracció transformen això en una instància de la classe adequada:

```
(defrule crear-persona-gran
    "Persona gran (>65)"
    (declare (salience 90))
    (fase (actual init))
    ?f <- (dades-solicitant (nom ?n) (edat ?e)
              (numeroPersones ?np) (pressupostMaxim ?pmax)
              ; ... més camps
    )
    (test (> ?e 65))
    =>
    (make-instance (sym-cat sol- (gensym*)) of PersonaGran
```

```
        (nom ?n) (edat ?e) (numeroPersones ?np)
        (pressupostMaxim ?pmax)
        ; ... més atributs
    )
    (retract ?f)
    (debug-print [ABSTRACCIO] Solicitant PersonaGran: ?n)
)
```

Tenim una regla similar per a cada categoria de solůlicitant, amb saliències diferents per establir prioritats. Per exemple, la regla de segona residència té saliència 100 (la més alta) perquè aquest criteri és més específic i ha de verificar-se primer.

Inferència de requisits:

Un cop tenim el solůlicitant classificat, inferim les seves necessitats:

```
(defrule abstraccio-familia-amb-fills
    "Les families amb fills puntuen escoles i zones verdes"
    (fase (actual abstraccio))
    ?sol <- (object (is-a Solicitant) (numeroFills ?fills))
    (test (> ?fills 0))
    (not (requisit-inferit (solicitant ?sol)
                           (categoria-servei ServeiEducatiu)))
    =>
    (assert (requisit-inferit
        (solicitant ?sol)
        (categoria-servei ServeiEducatiu)
        (obligatori no)
        (motiu "Familia amb fills necessita escoles")))
    (assert (requisit-inferit
        (solicitant ?sol)
        (categoria-servei ZonaVerda)
        (obligatori no)
        (motiu "Familia amb fills prefereix zones verdes")))
)
```

La condició `(test (> ?fills 0))` s'aplica a qualsevol subclasse de `Solicitant` que tingui fills, sigui `ParellaAmbFills` o `IndividuAmbFills`.

4.2.4 Fase de descart

Les regles de descart tenen totes una estructura similar:

```
(defrule resolucio-descartar-no-mascotes
    "Descartar si no permet mascotes i el solicitant en te"
    (fase (actual descart))
    ?sol <- (object (is-a Solicitant) (teMascotes si))
```

```
    ?of <- (object (is-a Oferta) (teHabitatge ?hab)
                   (disponible si))
    ?h <- (object (is-a Habitatge) (name ?hab)
                  (permetMascotes no))
    (not (oferta-descartada (solicitant ?sol) (oferta ?of)
       ))
    =>
    (assert (oferta-descartada
        (solicitant ?sol)
        (oferta ?of)
        (motiu "No permet mascotes")))
)
```

La condició `(not (oferta-descartada ...))` evita que la regla es dispari múltiples vegades per la mateixa oferta. Un cop s'ha creat el fet `oferta-descartada`, totes les regles posteriors que vulguin puntuar aquesta oferta fallaran perquè tenen la condició:

```
(not (oferta-descartada (solicitant ?sol) (oferta ?of)))
```

4.2.5 Fase de puntuació

Aquesta és la fase més complexa, amb desenes de regles que sumen o resten punts segons diferents criteris. Cada regla segueix aquest patró:

```
(defrule resolucio-puntuar-X
    "Explicació del criteri"
    (fase (actual scoring))
    ; Condicions sobre el solicitant
    ?sol <- (object (is-a TipusSolicitant) ...)
    ; Condicions sobre l'oferta/habitatge
    ?of <- (object (is-a Oferta) (teHabitatge ?hab) ...)
    ?h <- (object (is-a Habitatge) (name ?hab)
                  (caracteristica valor-desitjat))
    ; Recuperar la recomanació
    ?rec <- (Recomanacio (solicitant ?sol) (oferta ?of)
                         (puntuacio ?pts))
    ; No està descartada
    (not (oferta-descartada (solicitant ?sol) (oferta ?of)
       ))
    ; No hem aplicat ja aquest criteri
    (not (criteriAplicat (solicitant ?sol) (oferta ?of)
                         (criteri nom-criteri)))
    =>
    ; Modificar la puntuació
    (modify ?rec (puntuacio (+ ?pts PUNTS)))
    ; Registrar que hem aplicat el criteri
    (assert (criteriAplicat (solicitant ?sol) (oferta ?of)
                            (criteri nom-criteri)))
    ; Opcionalment, registrar el motiu
    (assert (punt-positiu (solicitant ?sol) (oferta ?of)
                          (descripcio "Text explicatiu")
```

```
                            (punts PUNTS)))
)
```

Listing 6: Patró de regla de puntuació

El template `criteriAplicat` és crucial per evitar que una mateixa regla es dispari múltiples vegades. Sense això, si un habitatge té terrassa i una regla suma 20 punts per tenir terrassa, aquesta regla es podria disparar infinitament perquè la condició es continua complint després de modificar la puntuació.

Exemple concret de regla de puntuació:

```
(defrule resolucio-puntuar-terrassa
    "Habitatge amb terrassa"
    (fase (actual scoring))
    ?sol <- (object (is-a Solicitant))
    ?of <- (object (is-a Oferta) (teHabitatge ?hab)
                   (disponible si))
    ?h <- (object (is-a Habitatge) (name ?hab)
                  (teTerrassaOBalco si))
    ?rec <- (Recomanacio (solicitant ?sol) (oferta ?of)
                         (puntuacio ?pts))
    (not (oferta-descartada (solicitant ?sol) (oferta ?of)
       ))
    (not (criteriAplicat (solicitant ?sol) (oferta ?of)
                         (criteri te-terrassa)))
    =>
    (modify ?rec (puntuacio (+ ?pts 20)))
    (assert (criteriAplicat (solicitant ?sol) (oferta ?of)
                            (criteri te-terrassa)))
    (assert (punt-positiu (solicitant ?sol) (oferta ?of)
                          (descripcio "Te terrassa o balco"
                             )))
)
```

4.2.6 Fase de classificació

Una vegada tenim totes les puntuacions calculades, és molt senzill assignar el grau:

```
(defrule classificacio-assignar-grau
    (fase (actual classificacio))
    ?rec <- (Recomanacio (solicitant ?sol) (oferta ?of)
                         (puntuacio ?pts) (grau NULL))
    (not (oferta-descartada (solicitant ?sol) (oferta ?of)
       ))
    =>
    (if (>= ?pts 70) then
        (modify ?rec (grau MoltRecomanable))
    else (if (>= ?pts 40) then
        (modify ?rec (grau Adequat))
    else (if (> ?pts 0) then
        (modify ?rec (grau Parcialment))
```

```
    )))
)
```

4.2.7 Fase de presentació

La presentació és més complexa perquè requereix ordenar les recomanacions i seleccionar el Top 3. Com que CLIPS no té funcions natives d'ordenació, hem implementat un bubble sort senzill dins de la pròpia regla:

```
(defrule presentacio-top3-per-solicitant
    (declare (salience -10))
    (fase (actual presentacio))
    ?sol <- (object (is-a Solicitant))
    =>
    (bind ?nom-sol (send ?sol get-nom))
    (printout t "SOLůLICITANT: " ?nom-sol crlf)

    ; Obtenir totes les recomanacions
    (bind $?recomanacions (create$))
    (do-for-all-facts ((?rec Recomanacio))
        (and (eq ?rec:solicitant ?sol) (neq ?rec:grau NULL))
        (bind $?recomanacions (create$ $?recomanacions ?rec))
    )

    ; Ordenar per puntuació (bubble sort)
    (bind ?n (length$ $?recomanacions))
    (if (> ?n 0) then
        (loop-for-count (?i 1 (- ?n 1))
            (loop-for-count (?j 1 (- ?n ?i))
                (bind ?rec1 (nth$ ?j $?recomanacions))
                (bind ?rec2 (nth$ (+ ?j 1) $?recomanacions))
                (if (< (fact-slot-value ?rec1 puntuacio)
                       (fact-slot-value ?rec2 puntuacio))
                    then
                    (bind $?recomanacions
                          (replace$ $?recomanacions ?j ?j ?rec2))
                    (bind $?recomanacions
                          (replace$ $?recomanacions (+ ?j 1)
                                    (+ ?j 1) ?rec1))
                )
            )
        )

        ; Mostrar només els 3 primers
        (bind ?max-mostrar (min 3 (length$ $?recomanacions
```

```
            )))
        (loop-for-count (?i 1 ?max-mostrar)
            (bind ?rec (nth$ ?i $?recomanacions))
            ; ... formatar i mostrar la recomanació
        )
    )
)
```

Listing 7: Ordenació i presentació del Top 3

4.3 Implementació de la metodologia de classificació heurística

La nostra implementació segueix fidelment la metodologia de classificació heurística, que és especialment adequada per a problemes on hem de categoritzar elements segons múltiples criteris.

4.3.1 Abstracció

El primer pas de la metodologia és obtenir una descripció abstracta del problema. En el nostre cas:

- **Dades brutes**: Edat, nombre de persones, fills, pressupost, etc.
- **Abstracció**: Classificació en perfils (PersonaGran, GrupEstudiants, ParellaAmbFills, etc.)
- **Inferència**: Deducció de necessitats implícites (requisits inferits)

Aquesta fase transforma dades heterogènies en conceptes del domini que es poden utilitzar en les regles posteriors.

4.3.2 Matching heurístic

En aquesta fase, emparellam cada oferta amb cada solůlicitant i avaluam el grau de coincidència. Ho fem en dues etapes:

Descart (matching binari): Algunes restriccions són absolutes: si no es compleixen, l'oferta es descarta directament. Això és equivalent a dir "aquestes ofertes no encaixen amb aquest solůlicitant de cap manera".

Puntuació (matching gradual): Per les ofertes que passen el filtre, calculam un grau de coincidència numèric que reflecteix quant de bé encaixen el solůlicitant i l'oferta. Això captura la idea que algunes ofertes són millors que altres, però no hi ha un tall net entre "boï "dolent".

4.3.3 Refinament

El refinament consisteix a transformar la puntuació numèrica en categories qualitatives:

- Molt Recomanable (70 punts)
- Adequat (40 punts)
- Parcialment Adequat (>0 punts)

Aquestes categories són més útils per a l'usuari que una puntuació numèrica crua, però la puntuació ens serveix per ordenar dins de cada categoria.

4.3.4 Justificació

Un aspecte fonamental de la classificació heurística és poder justificar les decisions. Ho hem implementat mitjançant els templates `punt-positiu` i `criteri-no-complert`, que s'omplen durant la fase de puntuació i es mostren després al generar la sortida.

4.4 Desenvolupament incremental i prototipatge

Un dels requisits de la pràctica era seguir una metodologia de desenvolupament incremental. Ho hem fet creant una sèrie de prototips de complexitat creixent.

4.4.1 Prototip 0: Estructura bàsica

El primer prototip no feia res útil, però establia l'estructura:

- Carregar l'ontologia
- Crear algunes instàncies de prova
- Una regla simple que imprimia "Sistema inicialitzat"

Aquest prototip ens va servir per validar que l'ontologia es carregava correctament i que podíem accedir als atributs dels objectes.

4.4.2 Prototip 1: Càlcul de proximitats i descart bàsic

El segon prototip ja feia alguna cosa útil:

- Calcular distàncies entre habitatges i serveis
- Classificar-les en molt a prop / distància mitjana / lluny
- Aplicar una sola regla de descart (pressupost estricte)
- Mostrar quines ofertes quedaven després del descart

Amb aquest prototip vam validar:

- La funció de càlcul de distàncies funcionava correctament
- Les regles podien accedir a localitzacions i coordenades
- El mecanisme de descart funcionava

4.4.3 Prototip 2: Classificació de solůlicitants

El tercer prototip afegia:

- Regles per classificar solůlicitants segons edat i situació familiar
- Funció interactiva per recollir dades del solůlicitant
- Creació automàtica de la instància de la classe adequada

Aquest va ser un punt d'inflexió important, perquè ara el sistema ja podia adaptar-se a diferents perfils.

4.4.4 Prototip 3: Inferència de requisits i puntuació bàsica

El quart prototip incorporava:

- Regles per inferir necessitats segons el perfil
- Un conjunt reduït de regles de puntuació (5-6 criteris)
- Classificació simple en graus de recomanació
- Sortida bàsica mostrant les millors ofertes

Amb aquest prototip ja teníem un sistema funcional de cap a cap, tot i que amb coneixement limitat.

4.4.5 Prototip 4: Expansió del coneixement

El cinquè prototip va ser una expansió massiva:

- Afegir més regles de descart (mascotes, accessibilitat, superfície, etc.)
- Afegir moltes més regles de puntuació (vam passar de 6 a més de 30)
- Diferenciar puntuacions segons el perfil del solůlicitant
- Registrar punts positius i criteris no complerts per a explicacions

Aquest va ser el prototip més laborios de desenvolupar, però el resultat va ser un sistema amb prou cobertura per a casos realistes.

4.4.6 Prototip 5: Presentació i refinament final

L'últim prototip es va centrar en la qualitat de la sortida:

- Implementar l'ordenació de recomanacions

- Seleccionar només el Top 3 per solůlicitant
- Formatar la sortida de manera clara i atractiva
- Mostrar explicacions detallades dels punts forts i aspectes a considerar
- Afegir estadístiques i missatges de debug opcionals

4.4.7 Beneficis del desenvolupament incremental

Aquest enfocament incremental ens ha aportat diversos avantatges:

Detecció primerenca d'errors: Cada prototip ens permetia validar que el que havíem implementat funcionava abans d'afegir més complexitat. Per exemple, vam descobrir que la nostra fórmula inicial de distància era incorrecta (utilitzàvem distància euclidiana en lloc de Manhattan) en el Prototip 1, abans de construir tota la lògica que depenia d'ella.

Iteració sobre el disseny: A mesura que anàvem implementant regles, ens adonàvem que alguns criteris que havíem pensat inicialment no tenien sentit o eren redundants. El prototipatge ens va permetre experimentar i ajustar sense haver de reescriure tot el sistema.

Divisió del treball: Els diferents membres de l'equip podíem treballar en paralůlel. Mentre un implementava més regles de puntuació, un altre podia estar refinant la presentació o afegint instàncies de prova.

Motivació: Tenir versions funcionals des del principi és molt més motivador que treballar durant setmanes en alguna cosa que no fa res fins al final. Cada prototip era un petit èxit que ens animava a continuar.

4.5 Aspectes destacables de la implementació

4.5.1 Funcions auxiliars

Hem creat diverses funcions auxiliars que encapsulen lògica reutilitzable:

```
(deffunction pregunta-si-no (?pregunta)
    "Pregunta amb resposta si/no"
    (printout t ?pregunta " (si/no): ")
    (bind ?resp (read))
    (while (and (neq ?resp si) (neq ?resp no))
        (printout t "Si us plau, respon 'si' o 'no': ")
        (bind ?resp (read))
    )
    ?resp
)

(deffunction pregunta-numero (?pregunta ?min ?max)
    "Pregunta amb resposta numerica amb limits"
    (printout t ?pregunta " [" ?min "-" ?max "]: ")
    (bind ?resp (read))
    (while (or (not (numberp ?resp))
```

```
                (< ?resp ?min) (> ?resp ?max))
        (printout t "Introdueix un numero entre "
                    ?min " i " ?max ": ")
        (bind ?resp (read))
    )
    ?resp
)
```

Listing 8: Funcions d'interacció amb l'usuari

Aquestes funcions fan que la funció principal `crear-perfil-solicitant` sigui molt més llegible i estructurada.

4.5.2 Mode debug

Hem implementat un sistema de missatges de debug que es pot activar/desactivar:

```
(defglobal ?*DEBUG* = TRUE)

(deffunction debug-print ($?msg)
   (if ?*DEBUG*
      then (printout t $?msg crlf)
   )
)
```

Així, durant el desenvolupament podem veure tots els passos intermedis:

```
[ABSTRACCIO] Solicitant de la categoria ParellaAmbFills: Garcia
[ABSTRACCIO] Garcia necessita escoles perque te fills
[DISTANCIES] hab-1 i escola-1 categoria Escola distancia: MoltAProp
[RESOLUCIO] DESCARTADA oferta-3 - No permet mascotes
[RESOLUCIO] PUNTUADA +20p A oferta-1 - Te terrassa
```

Però en producció podem desactivar-ho posant `?*DEBUG* = FALSE`.

4.5.3 Gestió de fases amb saliència

L'ús de saliències negatives per a les regles de transició de fase és elegant i efectiu. Garanteix que una fase no acaba fins que s'han disparat totes les regles d'aquella fase, sense necessitat de comptar explícitament quantes regles queden.

4.5.4 Evitar dispars múltiples amb criteriAplicat

El patró de crear un fet `criteriAplicat` cada vegada que una regla de puntuació es dispara és crucial per evitar cicles infinits. Sense això, modificar la puntuació no canvia les condicions de la regla (l'habitatge encara té terrassa), així que la regla es dispararia de nou infinitament.

Aquest patró és una bona pràctica en CLIPS quan fas modificacions incrementals a fets.

4.6 Dificultats trobades

Durant la implementació ens hem trobat amb diversos obstacles que val la pena documentar:

4.6.1 Gestió de la memòria

Amb moltes instàncies (6 habitatges, 10 serveis, 8 ofertes, etc.) i calculant totes les proximitats, es generen molts fets. En execucions llargues, això pot fer que CLIPS vagi lent. Hem mitigat això:

- Retracting facts que ja no necessitem (com `dades-solicitant` un cop creat el solůlicitant)
- Evitant crear fets redundants amb condicions `(not (...))`

4.6.2 Debugging de regles

Quan una regla no es dispara quan esperem, pot ser difícil saber per què. CLIPS ofereix `(matches regla-name)` per veure quines condicions fallen, però hem trobat més útil afegir `debug-print` dins de les regles per veure quan s'executen.

4.6.3 Ordenació en CLIPS

CLIPS no té funcions natives per ordenar llistes de facts, així que hem hagut d'implementar l'ordenació manualment. Hem escollit bubble sort per la seva simplicitat, tot i que no és l'algorisme més eficient. Per al nostre cas (ordenar 3-10 elements) és més que suficient.

4.6.4 Gestió de multislots

Els multislots en CLIPS són una mica peculiars. Per exemple, per afegir un element a un multislot has de fer:

```
(bind $?llista (create$ $?llista nou-element))
```

Aquesta sintaxi no és molt intuïtiva al principi i ens va costar una estona dominar-la.

4.7 Arquitectura de fitxers

Finalment, hem organitzat el codi en diversos fitxers per facilitar el manteniment:

- `ontologia.clp`: Definició de totes les classes (generat automàticament)
- `instancies_ciutat.clp`: Instàncies d'habitatges, serveis i ofertes
- `instancies_solicitants.clp`: Solůlicitants de prova predefinits

- `regles.clp`: Totes les regles del sistema, organitzades per fases
- `main.clp`: Funcions d'interacció i funció principal
- `presentacio.clp`: Regles específiques per a la presentació del Top 3
- `run.clp`: Script que carrega tot i executa el sistema

Aquest disseny modular fa que sigui fàcil localitzar i modificar parts concretes del sistema sense haver de navegar per un únic fitxer gegant.

5 Jocs de prova i Resultats

Per validar el sistema, s'han dissenyat jocs de prova que cobreixen els diferents perfils d'usuari identificats a la conceptualització. A continuació s'analitzen dos casos representatius que mostren com el sistema s'adapta a necessitats oposades.

5.1 Cas de Prova 1: Persona Gran (Sra. Montserrat)

Perfil: Dona de 72 anys, viu sola, necessita accessibilitat i transport públic. Pressupost 1500. Prioritza salut i silenci. **Resultats Obtinguts:**

- **Oferta Recomanada:** *oferta-avis-1* (Pis a l'Eixample).
- **Puntuació:** Alta (Molt Recomanable).
- **Justificació del Sistema:**
  - "Serveis de salut molt propers"(+25p).
  - "Transport públic molt a prop"(+25p).
  - "Te ascensor"(Requisit dur complert).
- **Descarts:** El sistema ha descartat correctament l'*oferta-parella-1* (Pis 4t sense ascensor a Gràcia) aplicant la regla `resolucio-descartar-no-accessible`. També ha descartat ofertes amb soroll alt a zones de festa.

5.2 Cas de Prova 2: Grup d'Estudiants (Grup UPC)

Perfil: 4 joves, pressupost ajustat per persona, necessiten pis moblat i prop de la universitat. **Resultats Obtinguts:**

- **Oferta Recomanada:** *oferta-estudiants-1* (Pis Zona Universitària).
- **Puntuació:** Molt Recomanable.
- **Justificació del Sistema:**
  - "A prop d'una universitat"(+20p).
  - "Ja moblat"(+25p).
  - "Dormitoris individuals per tothom"(+20p).
- **Descarts:** El sistema ha descartat automàticament tots els pisos que apareixen com `moblat no` mitjançant la regla `resolucio-descartar-estudiant-sense-mobles`, ja que l'expert (LLM) va indicar que els estudiants rarament compren mobles.

5.3 Cas de Prova 3: Família amb Fills

Perfil: Parella amb 2 fills (6 i 9 anys). Busquen espai i escoles. **Anàlisi:** El sistema valora positivament els pisos amb més de 100mš i amb 3 o més habitacions. Es dispara la inferència `abstraccio-familia-amb-fills` que afegeix la necessitat d'Escoles. El sistema puntua alt l'*oferta-familia-1* (Les Corts) perquè té escoles i parcs a menys de 500m.

5.4 Cobertura de les Proves

Els jocs de prova han validat:

1. **Restriccions Dures:** Pressupost estricte vs flexible, mascotes, ascensor.
2. **Inferència:** Detecció automàtica de necessitats (escoles, autopista per qui treballa fora).
3. **Càlcul Espacial:** Correcta classificació de distàncies (MoltAProp vs Lluny).
4. **Explicabilitat:** El sistema és capaç de llistar els motius positius i negatius.

6 Conclusions

6.1 Assoliment d'Objectius

En aquesta pràctica s'ha dissenyat i implementat amb èxit un Sistema Basat en el Coneixement per a la recomanació d'habitatges. S'han complert els objectius inicials:

- S'ha formalitzat el coneixement expert en una ontologia complexa amb classes, subclasses i relacions.
- S'ha utilitzat CLIPS per implementar un motor de regles modular (fases).
- El sistema és capaç de raonar sobre dades implícites (inferència de necessitats) i no només fer filtrat SQL.
- La integració del coneixement extret via LLM ha permès enriquir les regles amb criteris realistes (ex: importància del mobiliari per a estudiants).

6.2 Punts Forts del Sistema

1. **Modularitat:** La separació en fitxers d'instàncies, ontologia i regles, i l'ús de fases d'execució, fa que el sistema sigui fàcilment ampliable.
2. **Sensibilitat al Context:** El sistema tracta diferent la mateixa distància segons l'usuari. Una discoteca a prop és positiva per a un estudiant però negativa per a una família (regla d'evitació de serveis).
3. **Explicabilitat:** A diferència d'una caixa negra (com una xarxa neuronal), aquest sistema expert pot justificar exactament per què una oferta és bona, llistant els criteris aplicats.

6.3 Limitacions i Millores Futures

- **Càlcul de distàncies:** Actualment s'usa la distància euclidiana directa. Una millora seria integrar una API de mapes per calcular distàncies reals caminant o en transport públic.
- **Lògica Difusa (Fuzzy Logic):** Els límits de distància (500m) o preu són rígids. Implementar lògica difusa permetria una transició més suau (ex: "bastant a prop").
- **Dinamisme:** Les ofertes són estàtiques. En un entorn real, caldria connectar el sistema a una base de dades en temps real.

En conclusió, la pràctica ha permès aprofundir en el cicle de vida de l'enginyeria del coneixement, des de la conceptualització abstracta fins a la implementació concreta de regles de producció, demostrant la utilitat dels SBC en problemes de presa de decisions complexes.