UNIVERSITAT POLITÈCNICA DE CATALUNYA

Departament de llenguatges i sistemes informàtics

ACTUALITZACIÓ CONSISTENT DE BASES DE DADES DEDUCTIVES

Autor: Enric Mayol Sarroca
Director: Ernest Teniente i López

Barcelona, 2000

6. Arquitectura d'un mètode eficient

Al capítol 4 d'aquesta tesi hem definit formalment el mètode que proposem per a l'actualització consistent de bases de dades deductives. Com ja hem comentat anteriorment, en aquesta definició no s'han tingut en compte aspectes d'eficiència. És en aquest capítol, on introduirem les tècniques que ens permeten millorar l'eficiència del mètode proposat. Concretament, en aquest capítol, es presenta l'arquitectura d'una possible implementació del mètode per a actualitzar consistentment bases de dades deductives d'una forma més eficient.

A la secció 6.1, s'analitzen quines són les principals ineficiències que presentaria una implementació directa de les regles que defineixen el mètode (capítol 4) i, a la vegada, s'introdueix la idea general de les tècniques que proposem per a resoldre aquestes ineficiències. A les seccions 6.2 i següents, es presenten l'arquitectura general del mètode i una descripció detallada de cada un dels mòduls que la composen, posant especial atenció en la descripció de com s'assoleix la millora d'eficiència en l'actualització d'una vista i en el manteniment de les restriccions d'integritat.

6.1. Anàlisi de l'eficiència de la formalització del mètode

En la formalització del mètode del capítol 4, aquest es presenta com un procés que consisteix en l'alternança entre les derivacions constructives i les derivacions de consistència. El principal propòsit de la derivació constructiva és el d'obtenir un conjunt T d'esdeveniments bàsics que permeten satisfer la petició inicial d'actualització U. En canvi, en la derivació de consistència es comprova, mitjançant el conjunt auxiliar de condició C, que el conjunt T sigui consistent respecte la petició inicial U i respecte les restriccions d'integritat definides a la base de dades. En cada una d'aquestes derivacions es poden detectar millores respecte a l'eficiència de com realitzen la tasca encomanada.

6.1.1. Ineficiències en la derivació constructiva

En una derivació constructiva es poden identificar dos aspectes que són els causants principals de que aquest procés de derivació sigui especialment ineficient. En primer lloc, durant aquest procés de derivació es consideren regles d'esdeveniment de l'A(D) que no condueixen a cap solució vàlida de la petició d'actualització. Per altra banda, durant aquest mateix procés, es realitzen accessos a la base de dades que són redundants i molts d'ells es repeteixen diverses vegades.

Tal com es defineix una derivació constructiva, per tal d'obtenir totes les solucions a una petició d'actualització U en actualitzacions de fets bàsics, s'han de considerar totes les regles d'esdeveniment de la base de dades augmentada A(D) associades a cada un dels esdeveniments derivats que apareixen a la petició U. Com s'ha pogut comprovar en els exemples que s'han

mostrat en el capítol 4, no totes aquestes regles permeten obtenir un conjunt T. Això és degut a que cada regla d'esdeveniment és aplicable a algun estat concret de la base de dades. Quan en una branca de l'arbre de la derivació constructiva es selecciona un literal corresponent a un predicat de la base de dades, aquest literal pot no satisfer-se i, per tant, la branca que s'està considerant junt amb tota la feina realitzada en aquesta branca és rebutjada. Per altra banda, pot donar-se el cas de tenir que rebutjar la branca actual perquè s'ha seleccionat un literal que es correspon a un esdeveniment bàsic que no es pot incloure al conjunt T, perquè no es satisfan els prerequisits de la base de dades o perquè aquest esdeveniment és mútuament excloent amb altres esdeveniments del conjunt T. L'exemple següent mostra aquesta situació.

Exemple 6.1 Suposem la base de dades de l'exemple 3.1 i la petició d'actualització U = ιEmp(Marta, GM). En aquest cas, es tracta d'una petició d'actualització d'un fet derivat, i per tant, hem de considerar totes les seves regles d'esdeveniment per a obtenir-ne totes les possibles traduccions. Les regles d'esdeveniment de la A(D) d'aquest predicat derivat són les següents:

(I1) $\iota Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge \neg\delta Treb(\underline{p},c) \wedge \neg Aux1(\underline{p},c) \wedge \neg Aux9(\underline{p}) \wedge \iota Cont(\underline{p},c)$

(I2) $\iota Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge \neg\delta Treb(\underline{p},c) \wedge \neg Aux1(\underline{p},c) \wedge Cont(\underline{p},c1) \wedge \mu Cont(\underline{p},c1,c) \wedge c1 \neq c$

(I3) $\iota Emp(\underline{p},c) \leftarrow \neg Aux10(\underline{p}) \wedge \iota Treb(\underline{p},c) \wedge Cont(\underline{p},c) \wedge \neg\delta Cont(\underline{p},c) \wedge \neg Aux2(\underline{p},c)$

(I4) $\iota Emp(\underline{p},c) \leftarrow \neg Aux10(\underline{p}) \wedge \iota Treb(\underline{p},c) \wedge \neg Aux9(\underline{p}) \wedge \iota Cont(\underline{p},c)$

(I5) $\iota Emp(\underline{p},c) \leftarrow \neg Aux10(\underline{p}) \wedge \iota Treb(\underline{p},c) \wedge Cont(\underline{p},c1) \wedge \mu Cont(\underline{p},c1,c) \wedge c1 \neq c$

(I6) $\iota Emp(\underline{p},c) \leftarrow Treb(\underline{p},c1) \wedge \mu Treb(\underline{p},c1,c) \wedge c1 \neq c \wedge Cont(\underline{p},c) \wedge \neg\delta Cont(\underline{p},c) \wedge \neg Aux2(\underline{p},c)$

(I7) $\iota Emp(\underline{p},c) \leftarrow Treb(\underline{p},c1) \wedge \mu Treb(\underline{p},c1,c) \wedge c1 \neq c \wedge \neg Aux9(\underline{p}) \wedge \iota Cont(\underline{p},c)$

(I8) $\iota Emp(\underline{p},c) \leftarrow Treb(\underline{p},c1) \wedge \mu Treb(\underline{p},c1,c) \wedge c1 \neq c \wedge Cont(\underline{p},c2) \wedge \mu Cont(\underline{p},c2,c) \wedge c2 \neq c \wedge c1 \neq c2$

(A1) $Aux1(\underline{p},c) \leftarrow \mu Treb(\underline{p},c,c1)$

(A2) $Aux2(\underline{p},c) \leftarrow \mu Cont(\underline{p},c,c1)$

(A9) $Aux9(\underline{p}) \leftarrow Cont(\underline{p},c)$

(A10) $Aux10(\underline{p}) \leftarrow Treb(\underline{p},c)$

Observi's que en aquest cas, no existeix cap fet Treb(Marta,c) a la base de dades extensional, per tant, les regles I1, I2, I6, I7 i I8 no podran generar cap solució per la petició U. De forma similar, tampoc existeix cap fet Cont(Marta,c) a la base de dades i, per tant, les regles I3 i I5 no seran útils per obtenir una solució. Solament la regla I4 satisfà els requeriments de la base de dades per a obtenir una solució per la petició U.

En aquest cas, l'arbre de la derivació constructiva associat a l'objectiu ← ιEmp(Marta, GM) estarà composat per 8 branques, però solament en una d'elles assolirà la clàusula buida. Així

doncs, en aquest exemple, hem considerat 8 regles d'esdeveniment, però solament hem tingut èxit amb una d'elles (I4).

□

La definició de les regles d'esdeveniment de l'A(D) inclou literals que fan referència als predicats de la base de dades que defineixen el predicat derivat associat. Així doncs, en la construcció de l'arbre de derivació associat a una petició d'actualització U, es repeteixen consultes a la base de dades extensional que ja s'han realitzat en altres branques del mateix arbre de derivació i, fins i tot, es realitzen consultes a la base de dades extensional el resultat de les quals es podria deduir a partir del resultat de consultes realitzades amb anterioritat en altres branques. L'exemple següent mostra aquesta situació.

Exemple 6.2 Suposem la mateixa base de dades i petició U que a l'exemple 6.1. Per a generar l'arbre de la derivació constructiva associat a U, hem realitzat diverses consultes a la base de dades extensional. Sols tenint en compte els literals que fan referència al predicat bàsic Treb(p,c) les consultes a realitzar són les següents: en la branca associada a la regla I1, cal comprovar si el fet Treb(Marta,GM) és cert a la base de dades. Observi's que aquesta mateixa consulta cal repetir-la en la branca associada a la regla I2. De forma similar, la consulta Treb(Marta,x)? cal realitzar-la a les branques associades a les regles I6, I7 i I8; i la consulta ¬Treb(Marta,x)? es repeteix a les branques I3, I4 i I5.

Observi's que les consultes a la base de dades extensional que cal realitzar en aquest arbre de derivació es repeteixen en diferents branques. A més a més, algunes d'aquestes consultes es podrien deduir a partir del resultat de consultes anteriors, per exemple el resultat de les consultes ¬Treb(Marta,x)? i Treb(Marta,GM)? es pot deduir a partir del resultat de consultar Treb(Marta, x)?.

□

A partir d'aquests exemples, es pot veure que una implementació directa de les regles que defineixen la derivació constructiva conduiria a una implementació del nostre mètode força ineficient.

Per tal d'evitar aquesta manca d'eficiència i, en concret, per tal d'evitar considerar completament totes les regles de derivació i repetir consultes a la base de dades, proposem intentar determinar les consultes que cal realitzar a la base de dades extensional abans d'iniciar el procés de traducció de la petició d'actualització. Així doncs, pot n aconseguir realitzar una sola vegada les consultes a la base de dades i no repetir accessos innecessaris a la base de dades extensional. A més a més, el resultat d'aquestes consultes avançades, ens permetrà determinar a priori quina/es regla/es d'esdeveniment no poden ser utilitzades per a obtenir una solució T a la petició d'actualització i, per tant, no caldrà considerar-les durant procés de traducció. A la secció 6.5 presentem una tècnica que consisteix en fer una anàlisi de la petició d'actualització U per a determinar a priori l'extensió dels predicats bàsics i derivats que cal conèixer per a traduir la petició d'actualització U.

6.1.2. Ineficiències en la derivació de consistència

A l'igual que en la derivació constructiva, en la formalització de la derivació de consistència també es pot detectar una manca d'eficiència. En la derivació de consistència, el conjunt de condició C conté totes aquelles condicions que no poden ésser satisfetes per tal d'assegurar que el conjunt T sigui consistent. A cada incorporació d'un esdeveniment al conjunt T, es comprova que aquestes condicions no esdevinguin certes. Si alguna d'aquestes condicions esdevé certa, cal reparar-la afegint nous esdeveniments a T, i en cas de no poder-la reparar, es rebutja el conjunt T.

Tota condició del conjunt C és comprovada a cada nova incorporació d'un esdeveniment al conjunt T. Observi's que, en el cas que una condició sigui reparada, cal comprovar de nou totes les condicions del conjunt C. Les condicions del conjunt C es comproven independentment de si el nou esdeveniment afegit al conjunt T afecta realment o no la satisfactibilitat de la condició. Així doncs, una condició del conjunt C, pot ser comprovada (i reparada) diverses vegades durant tot el procés de comprovació de la consistència del conjunt T en diferents derivacions de consistència, i en molts casos, aquestes recomprovacions no detecten cap canvi en la satisfactibilitat de la condició.

Aquesta situació posa de rellevància que el procés de comprovació de la consistència del conjunt T pot ser especialment costós ja que el nombre de condicions a comprovar i el nombre de vegades que cal recomprovar cada condició poden ser elevats. En l'exemple següent es pot observar aquesta situació.

Exemple 6.3: Aquesta situació s'ha mostrat a l'exemple de la secció 4.2.4. Un cop generat el conjunt T={ιNumss(Mercè,800)}, es procedia a la comprovació de les restriccions d'integritat en una derivació de consistència (Figura 4.4). Cada una de les branques d'aquest arbre de derivació corresponia a la comprovació d'una regla d'esdeveniment d'inserció ιIc_i. Es comprovaven primer les restriccions d'integritat Ic1(p,c) i Ic2(p,c,s). Totes les branques associades fallaven de forma finita i s'acumulaven les condicions C1, C2, ... C11 al conjunt C. Al comprovar la tercera restricció d'integritat, la regla d'esdeveniment C13 esdevenia violada per T, i per tant, es reparava amb la incorporació a T de l'esdeveniment bàsic ιCont(Mercè,EDM). Al proposar aquest nou esdeveniment, calia comprovar de nou amb una derivació de consistència totes les condicions C1 ...C11 que ja havien estat comprovades prèviament. En aquest cas, la condició C2 esdevenia violada i es reparava amb l'esdeveniment ιEdat(Mercè). De nou, calia comprovar totes les condicions del conjunt C (C1 ...C13) i la resta de regles d'esdeveniment de la tercera restricció d'integritat.

En aquest exemple, es pot observar que l'ordre en que es comproven les condicions és rellevant per tal de reduir el nombre de recomprovacions que cal fer de cada condició. En aquest cas, s'han comprovat les condicions en el mateix ordre en que estan numerades (C1 fins C15). La condició C1 ha estat comprovada 3 cops i en cap cas ha estat violada. La condició C2 també

s'ha comprovat 3 cops i només en un cas ha estat violada i reparada. Les condicions C3...C13 s'han comprovat 2 cops i en cap cas s'han violat, excepte la C13 que s'ha violat i reparat en una ocasió. Les condicions C14 i C15 solament s'ha comprovat un cop i no han estat violades. En resum, podem veure que s'han comprovat un total de 30 condicions i sols s'han reparat 2 violacions.

□

Per tal de reduir el nombre de comprovacions que cal fer de les condicions del conjunt C proposem dues millores. En primer lloc, proposem no comprovar les condicions del conjunt de condició C després de cada canvi en el conjunt T, sinó retardar la seva comprovació fins al final del procés de traducció de la petició U. Un cop tenim tots els esdeveniments bàsics necessaris per a satisfer la petició d'actualització en el conjunt T, aleshores comprovem la consistència d'aquest conjunt T respecte a les condicions de C. D'aquesta manera, seguim assegurant que, al final del procés de comprovació de la consistència, cap de les condicions de C és violada per T, i ens podem aprofitar del fet que al final del procés de traducció, ja es coneixen tots (o quasi tots) els esdeveniments necessaris per a satisfer la petició d'actualització U. A més a més, en el cas de que la transacció T aconsegueixi satisfer totes les condicions del conjunt C, aconseguim reduir el nombre de vegades que cal comprovar cada una de les condicions del conjunt C. En el cas que la transacció T no pugui satisfer alguna d'aquestes condicions de C, llavors el nombre de condicions comprovades dependrà de l'ordre en que aquestes es comprovin.

En segon lloc, proposem definir un ordre específic de comprovació de les condicions del conjunt C analitzant les dependències entre aquestes condicions. Per a definir aquest ordre, tenim en compte el fet que un esdeveniment que pot reparar una condició que havia esdevingut certa pot, a la vegada, violar una segona condició. Així doncs, proposem que aquesta segona condició sigui comprovada un cop s'hagi assegurat la falsedat de la primera. Aquest ordre de comprovació entre condicions es determina a partir de la informació representada en el que anomenem Graf de Precedències.

A la secció 6.3 es presenta la definició i el mecanisme per a construir el Graf de Precedències, mentre que a la secció 6.4, es mostra com es pot utilitzar aquest graf per a mantenir de forma eficient, la consistència del conjunt T respecte al conjunt de condicions C.

6.2 Arquitectura general del mètode

En aquesta secció es descriu l'arquitectura general d'una possible implementació del mètode definit al capítol 4. Aquesta arquitectura i la seva descomposició modular estan directament determinats per les tècniques de millora de l'eficiència que s'han comentat a la secció anterior i que s'explicaran en més detall en les seccions següents.

En les seccions anteriors, i especialment en la definició formal del mètode, hem fet referència a la petició d'actualització U, al conjunt T i al conjunt de condicions C. En la descripció de

l'arquitectura de la implementació del mètode, també farem referència a aquests conceptes, però en aquest cas, farem referència a una traducció TC, la qual està composada per una transacció T i el conjunt de condicions C associat.

Definició 6.1: Una *traducció TC* d'una petició d'actualització U està composada per una transacció T i un conjunt addicional de condicions C. La *transacció T* conté les actualitzacions de fets bàsics necessàries per a satisfer la petició U, i el *conjunt de condicions C* conté aquelles condicions que asseguren que la transacció T realment satisfà la petició U.

Les transaccions T que satisfan les condicions del conjunt associat C són traduccions correctes de la petició d'actualització U. A més a més, les transaccions T que no violin cap restricció d'integritat ni cap condició del conjunt C són solucions a la petició d'actualització U.

Definició 6.2: Sigui D una base de dades deductiva, U una petició d'actualització i TC una traducció de la petició U composada per la transacció T i el conjunt de condicions C, aleshores la transacció T és una *solució* de la petició d'actualització U si i sols si la transacció T no viola cap condició del conjunt C ni cap restricció d'integritat definida a la base de dades D.

Definició 6.3: Siguin S una solució a una petició d'actualització U, aleshores podem dir que S és una *solució mínima* si i sols si no existeix cap altre solució S_i tal que $S_i \subseteq S$.

L'arquitectura del mètode l'hem dividida en dos components: un component que s'executarà en temps de definició i un component a executar en temps d'execució. A la Fig.6.1 es presenta el component de l'arquitectura del mètode en temps de definició. Aquest està composat per dos mòduls: el Generador de la Base de Dades Augmentada A(D) i el Generador del Graf de Precedències.

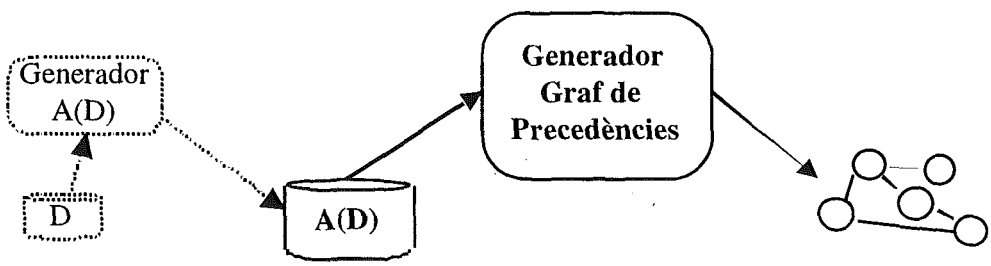


Fig. 6.1 Arquitectura del mètode en temps de definició

El Generador de l'A(D) és el mòdul encarregat de la generació de la Base de Dades Augmentada A(D) a partir de l'esquema d'una base de dades deductiva D. Aquest mòdul queda en part fora de l'àmbit del treball que es presenta en aquesta tesi, ja que es correspon al treball presentat a [Urp93]. La única part d'aquest mòdul que sí és específica del nostre treball són les reescriptures de les regles d'esdeveniment que s'han introduït al capítol 3 d'aquesta tesi.

El mòdul Generador del Graf de Precedències és l'encarregat de generar un Graf de Precedències entre condicions. En aquest graf es representen com a nodes les condicions que permetran assegurar, en temps d'execució, que tota traducció T és consistent i no viola cap restricció d'integritat. Les arestes del graf estableixen les precedències entre aquestes

condicions. En temps d'execució, aquestes precedències permetran determinar l'ordre en que cal comprovar les condicions. En el procés de generació del Graf de Precedències entre condicions, tant sols es té en compte la informació sintàctica de la Base de Dades Augmentada A(D). Així doncs, aquest Graf de Precedències es pot generar en temps de compilació, ja que és independent de la petició d'actualització U i dels fets de la base de dades extensional EDB.

A la Fig.6.2 es presenta el component de l'arquitectura del mètode en temps d'execució. Aquest està composat per dos mòduls: el mòdul de Manteniment de la Consistència i el mòdul d'Actualització de Vistes.

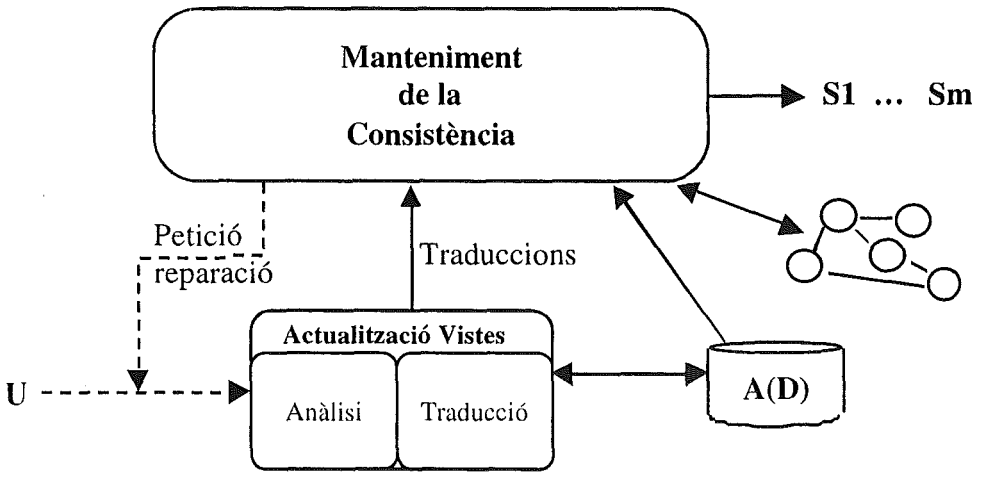


Fig. 6.2 Arquitectura del mètode en temps d'execució

Donada una petició d'actualització U proporcionada per l'usuari, el mètode s'inicia amb l'execució del mòdul d'Actualització de Vistes. En el cas que la petició U només inclogui peticions d'actualitzar predicats bàsics, llavors el Mòdul d'Actualització de Vistes genera una sola traducció TC on T=U i C=∅. En el cas que la petició U contingui actualitzacions de vistes o actualitzacions a evitar, llavors el mòdul d'Actualització de Vistes obtindrà totes les traduccions associades a la petició U. En qualsevol dels dos casos anteriors, per a cada traducció TC es comprovarà la seva consistència cridant al mòdul de Manteniment de la Consistència. En cas de que el mòdul de Manteniment de la Consistència requereixi reparar alguna condició mitjançant una actualització d'una vista, es cridarà al mòdul d'Actualització de Vistes perquè obtingui totes les traduccions a la petició de reparació.

El mòdul de Manteniment de la Consistència és l'encarreg d'assegurar la consistència de cada una de les solucions generades per una petició d'actualització U. D'aquesta forma assegurem que tota solució S no viola cap restricció d'integritat i que, a la vegada, satisfà realment la petició inicial d'actualització U.

En aquest procés de manteniment de la consistència, proposem la utilització del Graf de Precedències per a determinar l'ordre en que cal comprovar les restriccions d'integritat i les condicions del conjunt C. Quan s'ha comprovat que no resta cap restricció ni condició violada, les actualitzacions bàsiques de la transacció T composaran la solució S.

El resultat final del mòdul de Manteniment de la Consistència és el conjunt de totes les solucions mínimes que satisfan la petició inicial U i no violen cap restricció d'integritat.

El mòdul d'Actualització de Vistes és l'encarregat d'obtenir totes les traduccions a una petició d'actualització d'un predicat derivat. Aquesta petició pot ser requerida per l'usuari o pot correspondre a una petició de reparació d'alguna condició o restricció d'integritat.

Aquest mòdul està dividit en dos submòduls: el mòdul d'Anàlisi i el mòdul de Traducció. En el primer d'ells es realitza un treball preparatori per tal de millorar l'eficiència del procés de traducció. A partir d'una petició d'actualització U es determinen quines consultes a la base de dades serà necessari fer per a traduir aquesta petició. Es realitzen aquestes consultes i, a partir del seu resultat, es determinen quines regles de la A(D) permetran generar alguna traducció a la petició d'actualització. Aquestes regles s'especialitzen de forma que al ser avaluades en l'etapa de Traducció no es requereixin accessos addicionals a la base de dades extensional. Després, el mòdul de Traducció s'encarrega d'obtenir totes les traduccions a la petició d'actualització tant sols considerant les regles d'esdeveniment especialitzades. Així doncs, en el procés d'Actualització de Vistes, i gràcies al treball preparatori realitzat a l'etapa d'Anàlisi, es tradueix una petició d'actualització considerant únicament les regles d'esdeveniment que ens permetran generar alguna traducció i tant sols s'accedeix una única vegada als fets de la base de dades extensional necessaris per a realitzar aquesta traducció.

El resultat obtingut del mòdul d'Actualització de Vistes és un conjunt de totes les traduccions a una petició d'actualització U. Cada una d'aquestes traduccions està composada per una transacció T que conté les actualitzacions de fets bàsics necessàries per a satisfer la petició, i el conjunt de condició C que ens permet assegurar que aquesta transacció satisfà realment la petició U.

Com es pot comprovar, hi ha un cert paral·lelisme entre els mòduls en temps d'execució i les Derivacions Constructiva i de Consistència de la formalització del mètode. El mòdul d'Actualització de Vistes bàsicament implementa el procés associat a una Derivació Constructiva, és a dir, a partir d'una petició d'actualització obté els diferents conjunts d'actualitzacions bàsiques que permeten satisfer-la. A part de les millores a nivell d'eficiència que s'incorporen en aquest mòdul, la principal diferència està en que les condicions del conjunt de condició C no es comproven en cada incorporació d'un esdeveniment al conjunt T, sinó que es deixen per a ser comprovades totes juntes al final, en el Mòdul de Manteniment de la Consistència. En aquest sentit, les traduccions obtingudes al final d'aquest mòdul no es pot assegurar que siguin consistents. El mòdul de Manteniment de la Consistència és l'encarregat de la imposició d'aquestes condicions del conjunt C i d'assegurar que no es viola cap restricció d'integritat. Aquest darrer mòdul es correspon a una implementació de la Derivació de Consistència

L'alternança entre les Derivacions Constructiva i de Consistència segueix existint en aquesta arquitectura del mètode encara que no és tan freqüent com en la formalització del mètode. Si la petició de l'usuari inclou alguna actualització d'un predicat derivat, el mòdul d'Actualització de Vistes s'executa en primer lloc i després es requereix l'execució del mòdul de Manteniment de la Consistència un sol cop al finalitzar l'execució del mòdul d'Actualització de Vistes, en lloc d'executar-se, com li correspondria, després de cada nova incorporació al conjunt T. Per altra banda, el mòdul de Manteniment de la Consistència sí que pot requerir en diferents moments l'execució del mòdul d'Actualització de Vistes com un subprocés per a reparar alguna condició.

A les seccions 6.3, 6.4 i 6.5 s'explica amb més detall cada un dels mòduls que composen l'arquitectura del mètode proposat.

6.3 Mòdul Generació del Graf de Precedències

El Graf de Precedències és un graf dirigit en el que es representen, com a nodes, les diferents condicions que asseguren la consistència d'una transacció. Les arestes d'aquest graf estan etiquetades amb un esdeveniment i estableixen les precedències entre aquestes condicions. Les precedències entre condicions serviran, en temps d'execució, per a determinar quines condicions comprovar i en quin ordre comprovar-les.

Una transacció T, per a ser considerada solució d'una petició d'actualització U, ha de permetre que en a la base de dades un cop actualitzada no es violi cap restricció d'integritat i se satisfaci realment la petició inicial d'actualització U. Així doncs, les dues causes per les que una traducció T pot no ser considerada solució són, per una banda, perquè la seva aplicació viola alguna restricció d'integritat, i per l'altra, perquè indueix canvis que impedeixen satisfer la petició inicial d'actualització U.

Aquestes dues situacions es detecten amb el nostre mètode perquè, en el primer cas, s'indueix la inserció del fet del predicat d'inconsistència Ic, i en el segon cas, perquè es viola alguna condició del conjunt de condició C.

Tenint en compte com es detecten les violacions de les restriccions d'integritat, podem considerar el cos de les regles d'esdeveniment d'inserció dels predicats d'inconsistència definits a la base de dades augmentada (ιIc_i), com a les condicions que s'han de satisfer per a violar una restricció d'integritat. A més a més, aquestes condicions, al igual que les condicions del conjunt C, representen situacions que tota transacció T ha d'evitar per a ser considerada una solució. Llavors, podem representar i tractar de forma similar les condicions associades a les restriccions d'integritat i les condicions associades al conjunt C. Podem generalitzar definint una Condició com a un objectiu que estableix una situació que tota transacció T ha d'evitar.

Definició 6.4: Una *condició* expressa una situació, en forma denegada, que no ha de ser mai satisfeta per cap transacció T per tal que aquesta T pugui ser considerada solució a una petició d'actualització.

Exemple 6.4: Suposem per exemple les condicions següents:

$\leftarrow \delta Cont(\underline{Enric}, UPC)$

$\leftarrow Sou(\underline{p},c,s) \wedge \neg\delta Sou(\underline{p},c,s) \wedge \neg Aux6(\underline{p},c,s) \wedge Treb(\underline{p},c) \wedge \delta Treb(\underline{p},c)$

$Aux6(\underline{p},c,s) \leftarrow \mu Sou(\underline{p},c,s,c1,s1)$

La primera condició estableix que no es pot esborrar el fet que l'Enric té un contracte amb la companyia UPC. En canvi la segona condició descriu una forma de violar la segona restricció d'integritat de l'exemple 3.1 que cal evitar. És a dir, no es pot donar el cas de tenir una persona 'p' que treballa en una companyia 'c' en la que cobra un sou de 's' Euros, i que una transacció T l'esborri com a treballador d'aquesta companyia sense esborrar-li o modificar-li el sou. Observi's que aquestes condicions es representen en forma de denegació i poden estar completament o parcialment instanciades.

□

Encara que l'estructura i el tractament de les condicions serà bàsicament el mateix, l'origen de les condicions és diferent. Podem distingir dos tipus de condicions segons el seu origen: les Condicions de Restriccions d'Integritat, que asseguren que tota transacció no viola cap restricció d'integritat, i les Condicions d'Actualització de Vistes, que asseguren que una transacció T satisfà una petició inicial d'actualització U. Els dos tipus de condicions tenen la mateixa estructura i es tractaran de forma similar. La única diferència rau en que les Condicions de Restriccions d'Integritat han de ser satisfetes per tota transacció que es vulgui aplicar sobre la base de dades, en canvi, les Condicions d'Actualització de Vistes estan sempre associades a una transacció obtinguda com a traducció d'una petició d'actualització d'un predicat derivat.

Definició 6.5: Una *Condició de Restricció d'Integritat* està definida pel cos de la regla d'esdeveniment d'inserció del predicat d'inconsistència, associat a la restricció d'integritat. Per tant, aquestes condicions són conegudes en temps de definició. Aquestes condicions han de ser comprovades per tota transacció que es vulgui aplicar sobre la base de dades.

La resta de condicions són Condicions d'Actualització de Vistes.

Definició 6.6: Una *Condició d'Actualització de Vistes* és una condició del conjunt de condicions C obtinguda en un procés d'actualització de vistes. Aquestes condicions són conegudes i generades en temps d'execució i són específiques per a les transaccions que han de satisfer una petició inicial d'actualització U.

El fet de no conèixer en temps de definició totes les possibles condicions que haurà de satisfer una solució a una petició d'actualització U, no permet generar en temps de definició un graf que inclogui totes aquestes condicions i permeti definir l'ordre en que cal comprovar-les. Però una anàlisi de l'origen de les Condicions d'Actualització de Vistes permet observar que aquestes condicions segueixen uns patrons molt concrets. Totes elles s'han obtingut a partir de l'avaluació i la instanciació (parcial) de condicions més generals. Aquestes condicions més generals són d'un d'aquests tres tipus:

- Condicions que imposen la no ocurrència d'algun esdeveniment bàsic. En aquest cas, el cos de la condició contindrà un sol literal que es correspon a aquest esdeveniment bàsic.
- Condicions que imposen la no inducció d'algun esdeveniment derivat. En aquest cas, tindrem tantes condicions com regles d'esdeveniment hi hagi definides per aquest esdeveniment en la base de dades augmentada A(D). Cada condició es correspon al cos d'una d'aquestes regles d'esdeveniment.
- Condicions que imposen la no inducció d'algun predicat auxiliar o predicat de transició. De la mateixa manera que en el cas anterior, tindrem tantes condicions com regles defineixin aquests predicats. El cos de cada una d'aquestes regles es correspondrà a una condició diferent.

Tenint en compte aquesta generalització, és possible definir completament i en temps de definició, un Graf de Precedències. Les condicions que associarem als nodes del Graf de Precedències seran les Condicions de Restriccions d'Integritat i aquestes Condicions d'Actualització de Vistes més generals. Totes elles estan definides independentment del contingut de la base de dades extensional i de la petició d'actualització U. En temps d'execució, les condicions del conjunt C generades en un procés d'actualització de vistes s'associaran al node de la condició més general que li correspongui.

Per a identificar cada una de les condicions i etiquetar els nodes del Graf de Precedències, els hi associarem un identificador. Cada condició s'identificarà per l'identificador de la regla d'esdeveniment associada de la base de dades augmentada A(D). Així doncs, les Condicions de Restriccions d'Integritat tindran un identificador del tipus C_i. Les Condicions d'Actualització de Vistes, tindran un identificador I_i, D_i, M_i o A_i segons estiguin associades a regles d'esdeveniment d'inserció, d'esborrat, de modificació, o a predicats auxiliars, respectivament. Les Condicions d'Actualització de Vistes que restringeixen l'ocurrència d'un esdeveniment bàsic s'identificaran amb un identificador del tipus B_i.

Exemple 6.5: En aquest exemple es mostren algunes de les condicions, i els identificadors que els hi associem tenint en compte el seu origen.

(C1) $\leftarrow Emp(\underline{p},c) \wedge \neg\delta Emp(\underline{p},c) \wedge \neg Aux3(\underline{p},c) \wedge Edat(\underline{p}$` $\delta Edat(\underline{p})$

(C2) $\leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Edat(\underline{p}) \wedge \neg\iota Edat(\underline{p})$

(C3) $\leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(C4) $\leftarrow Emp(\underline{p},c) \wedge \mu Emp(\underline{p},c,c1) \wedge c1 \neq c \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(B4) $\leftarrow \mu Treb(\underline{Pau},Za,c1)$

(I11) $\leftarrow \neg Aux11(\underline{Pere}) \wedge Treb(\underline{Pere},c1) \wedge \mu Treb(\underline{Pere},c1,Bc) \wedge c1 \neq Bc \wedge \iota Cont(\underline{Pere},Bc)$

(A3) $Aux3(\underline{p},c) \leftarrow \mu Emp(\underline{p},c,c1)$

(A11) $Aux11(\underline{p}) \leftarrow Emp(\underline{p},c)$

Les condicions C1...C4 són les condicions associades a la primera restricció d'integritat (Ic1). La condició d'actualització de vistes B4 restringeix l'ocurrència d'un esdeveniment bàsic. La darrera condició és una condició d'actualització de vistes obtinguda a partir de l'avaluació de la condició general associada a la regla d'esdeveniment I11.

□

A partir d'aquest moment, quan ens referim a una condició, ho farem independentment del seu origen. En cas de ser necessari diferenciar el tipus de condició, ens referirem a ella com a Condició de Restricció d'Integritat o com a Condició d'Actualització de Vistes.

A part dels nodes, els altres elements que composen el Graf de Precedències són les arestes entre nodes. Per a determinar quines són les precedències entre els nodes (condicions) del graf, cal conèixer quin són els esdeveniments que poden violar i/o reparar cada una d'aquestes condicions. A més a més, per a determinar aquests esdeveniments, cal identificar prèviament quines condicions i esdeveniments derivats es poden induir per l'ocurrència de determinats esdeveniments bàsics. Tota aquesta informació es pot obtenir tenint únicament en compte la base de dades augmentada A(D), i per tant, es pot determinar en temps de definició.

Així doncs, abans de definir formalment el Graf de Precedències i el mecanisme per a obtenir-lo, introduirem alguns conceptes preliminars. A la secció 6.3.1 introduïm, entre altres, el concepte de Graf de Dependències entre esdeveniments [Cos95]. Aquest graf permet determinar quins esdeveniments i condicions depenen de que quins esdeveniments bàsics. A la secció 6.3.2 es defineix el concepte de Precedència entre condicions i a la secció 6.3.3 es presenten algunes optimitzacions que s'apliquen per a simplificar el conjunt de precedències identificades. A la secció 6.3.4 es defineix i es descriu com es genera el Graf de Precedències. Finalment, a la secció 6.3.5 es comenten algunes alternatives a la definició i al procediment de generació del Graf de Precedències.

6.3.1 Graf de Dependències entre esdeveniments

A l'aplicar una transacció composada per esdeveniments bàsics sobre la base de dades, es pot induir l'ocurrència de diferents esdeveniments derivats i la violació d'alguna condició. Aquests fets induïts i les condicions violades es poden determinar a partir de les regles d'esdeveniment de la base de dades augmentada A(D).

Exemple 6.6: Suposem la base de dades de l'exemple 3.1 i una transacció T composada pels esdeveniments T={ ιTreb(Anna,UPC), ιCont(Anna,UPC)}. Aquesta transacció permet induir els fets derivats ιEmp(Anna,UPC), ιActiu(Anna) i ιContractat(Anna) mitjançant les regles d'esdeveniment I4, I10 i I13 respectivament. Però també permet induir la violació de la condició de restricció d'integritat C2 de l'exemple 6.5.

□

Així doncs, utilitzant la definició de la base de dades augmentada A(D) es pot identificar quins efectes tenen l'ocurrència d'un conjunt d'esdeveniments bàsics sobre els esdeveniments

derivats i condicions. Aquests efectes es determinen a partir de l'anàlisi de les dependències entre esdeveniments i condicions.

Definició 6.7: Sigui E un esdeveniment i C una condició o un esdeveniment derivat. Podem dir que *C depèn directament de E* si: en cas de ser C un esdeveniment derivat, existeix una regla a la A(D) tal que en el cos de la regla hi apareix l'esdeveniment E i C apareix en el cap de la regla; en el cas en que C és una condició, E apareix en el cos d'aquesta condició.

Definició 6.8: Sigui E un esdeveniment i C una condició o un esdeveniment derivat. Si C depèn directament de E i l'esdeveniment E apareix com a literal positiu (resp. negatiu) en el cos de la regla que defineix C, llavors diem que la dependència és *positiva* (resp. *negativa*).

Observi's que es pot donar la situació que una condició o esdeveniment C depengui directament de forma positiva i negativa d'un mateix esdeveniment E.

Exemple 6.7: Consideri's la condició C següent: $\leftarrow$ Treb(p,c) $\wedge$ μTreb(p,c,c1) $\wedge$ c1$\neq$c $\wedge$ Cont(p,c) $\wedge$ $\neg\mu$Cont(p,c,c1). Existeix una dependència positiva (negativa) entre aquesta condició i l'esdeveniment E = μTreb(p,c,c1) (μCont(p,c,c1)), ja que C depèn directament de E i aquest esdeveniment apareix com a literal positiu (negatiu) en el cos de la condició. □

Exemple 6.8: Consideri's l'esdeveniment derivat C = δEmpl(p,c) i totes les seves regles d'esdeveniment associades (D1, D2, D3, D4). En aquest cas, podem identificar una dependència positiva entre C i els esdeveniments E del conjunt { δTreb(p,c), μTreb(p,c,c1), δCont(p,c), μCont(p,c,c1)}. Per altra banda, existeix una dependència negativa entre C i els esdeveniments E' del conjunt { μTreb(p,c,c1), μCont(p,c,c1)}. Observi's que en aquest mateix exemple l'esdeveniment δEmpl(p,c) depèn de forma positiva i negativa dels esdeveniments bàsics μTreb(p,c,c1) i μCont(p,c,c1). □

Tenint en compte totes aquestes dependències de condicions i esdeveniments respecte altres esdeveniments podem construir el Graf de Dependències entre esdeveniments [Cos95].

Definició 6.9: Donat un conjunt d'esdeveniments i de condicions, un *Graf de Dependències* GD és un parell GD=<N, A> on N es un conjunt finit de nodes i, A$\subseteq$NxN és el conjunt de arestes dirigides entre nodes. Cada node n$\in$N està etiquetat amb l'identificador d'una condició o d'un esdeveniment. Donats dos nodes n i n', existeix una aresta a=(n,n') si i sols si el node n' depèn directament del node n. Les arestes estan marcades positivament (negativament) si la dependència és positiva (negativa).

Exemple 6.9: En aquest exemple, per a fer més visible del Graf de Dependències entre esdeveniments, hem considerat les Condicions de Restriccions d'Integritat (C1 , ... C15), però solament algunes de les condicions d'Actualització de Vistes (D1, D2, D3, D4, M1). El Graf de Dependències d'aquest exemple es mostra a la Figura 6.3. Observi's que, encara que no

apareixen en la figura, per a cada una de les condicions identificades per Bi, existeix una dependència positiva entre aquesta condició i l'esdeveniment bàsic que la defineix. □

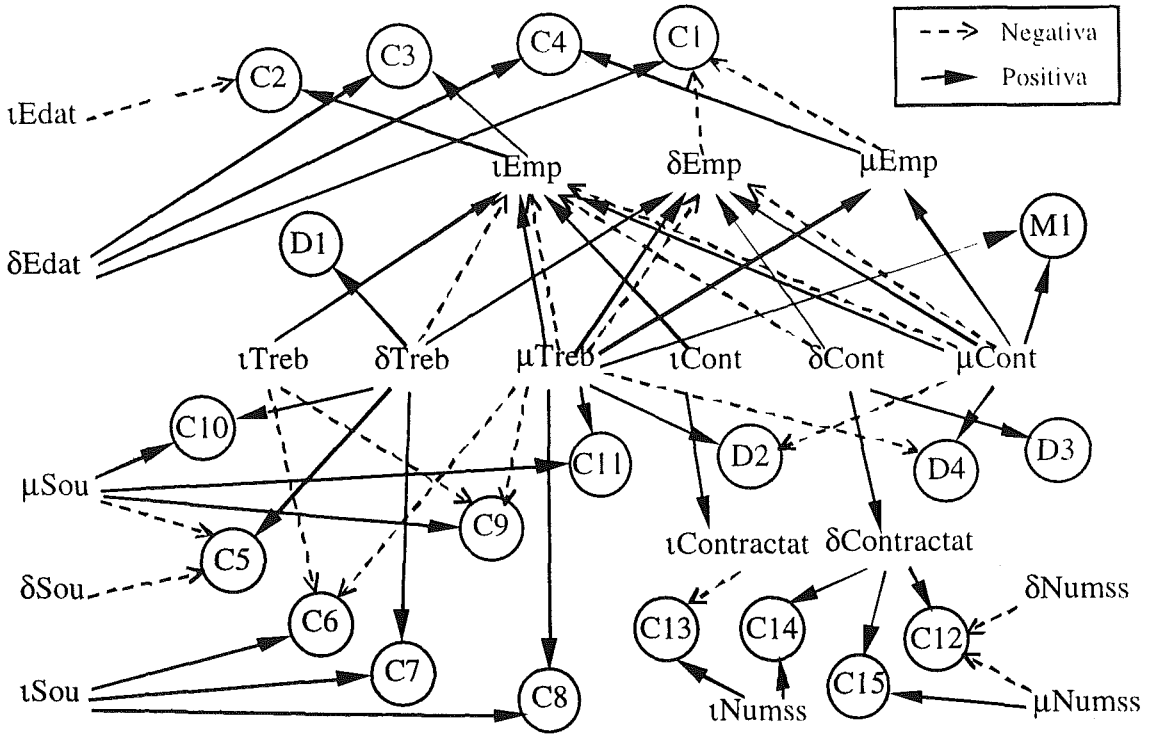


Fig. 6.3. Graf de Dependències de l'exemple 6.9

A partir de la informació representada en el Graf de Dependències entre esdeveniments es poden determinar altres tipus de dependències entre esdeveniments.

Definició 6.10: Sigui GD un Graf de Dependències entre esdeveniments, n i n' dos nodes qualssevol d'aquest graf. Aleshores podem dir que:

- *n depèn de n'* si existeix a GD un camí des de n' fins a n.
- *n depèn de forma parell (senar) de n'* si existéix a GD un camí des de n' fins a n amb un nombre parell (senar) d'arestes negatives.

Les dependències que existeixen entre condicions i esdeveniments permeten determinar els violadors potencials i reparadors potencials de les condicions. Intuïtivament, un violador potencial d'una condició C és un esdeveniment bàsic que, aplicat sobre la base de dades, pot fer la condició certa. En canvi, un reparador potencial d'una condició C és un esdeveniment bàsic que, aplicat sobre la base de dades, pot fer falsa aquesta condició. Aquesta informació es pot identificar analitzant sintàcticament les dependències explícites representades en el Graf de Dependències.

Definició 6.11: Sigui C una condició i E un esdeveniment bàsic.

a) E és un *violador potencial* de C si C depèn de forma parell de E.

b) E és un *reparador potencial* de C si C depèn de forma senar de E.

Observi's que en la definició anterior es pot donar el cas que un esdeveniment bàsic E sigui a la vegada un violador i un reparador potencial de la mateixa condició. Aquesta situació apareix quan en el Graf de Dependències existeixen dos camins alternatius des de E fins a C, un amb un nombre parell d'arestes negatives i, l'altre camí amb un nombre senar d'arestes negatives.

Exemple 6.10: Tenint en compte el Graf de Dependències entre esdeveniments de la figura 6.3, és fàcil comprovar que l'esdeveniment δTreb és un violador potencial de la condició C5, ja que el camí entre el node C5 i el node δTreb conté 0 arestes negatives. En canvi, existeixen dos reparadors potencials per aquesta condició: μSou i δSou. Els camins entre el node C5 i els nodes μSou i δSou contenen una sola aresta negativa. De la mateixa manera, es pot comprovar que els violadors potencials de la condició C2 són els esdeveniments ιTreb, μTreb, ιCont i μCont. En canvi els reparadors potencials són ιEdat, δTreb, μTreb, δCont i μCont. En aquest cas, l'esdeveniment μTreb (i el μCont) es considerarà un violador i reparador potencial de la condició C2.

□

En temps de definició no es pot assegurar que un esdeveniment violi realment una condició, ja que per a violar-se es requereix la satisfacció de la resta de literals que defineixen la condició, i aquesta informació sols es pot conèixer en temps d'execució. De manera similar, un esdeveniment no sempre repara totalment una condició, ja que aquesta pot estar violada per diferents esdeveniments, i per a reparar-la, caldrà reparar totes les seves violacions amb l'aplicació de més d'un esdeveniment reparador. Aquestes són les causes perquè en temps de definició sols podem parlar de violadors i reparadors potencials. Serà, en temps d'execució, quan es comprovi realment si una condició està violada i quins esdeveniments la poden reparar.

Es pot donar el cas que, per a alguna condició, no existeixi cap reparador potencial, ja que no depèn de forma senar de cap esdeveniment bàsic. Tenint únicament en compte informació sintàctica de la condició, es poden diferenciar dos tipus de condicions: aquelles condicions que al ser violades no es podran reparar en cap cas, ja que no tenen cap reparador potencial associat; i les condicions que al violar-se podrien ser reparades proposant noves actualitzacions de predicats bàsics. A aquestes condicions les anomenem, respectivament, Condicions de Comprovació i Condicions de Generació. Tota condició C pot classificar-se únicament en un sol d'aquests dos tipus.

Definició 6.12: Sigui C una condició. C és una *Condició de Comprovació* si no té associat cap reparador potencial. C és una *Condició de Generació* si té associat algun reparador potencial.

Exemple 6.11: Les condicions C4, C7, C8, C10, C11, C14, C15, D1, D3 i M1 representades en el Graf de Dependències de la figura 6.3 són Condicions de Comprovació ja que no depenen de forma senar de cap esdeveniment bàsic. La resta de condicions són Condicions de Generació. Existeix un grup de Condicions d'Actualització de Vistes que sempre

són condicions de Comprovació. Aquestes són les condicions amb l'etiqueta Bi que tenen la següent forma ←Ev (amb Ev esdeveniment bàsic). Observi's que aquestes condicions tenen un únic violador potencial (l'esdeveniment Ev) i no tenen cap reparador potencial.

□

La distinció entre Condicions de Restriccions d'Integritat i Condicions d'Actualització de Vistes és ortogonal a la classificació entre Condicions de Comprovació i Condicions de Generació. En la primera classificació es considera quin és l'origen d'aquestes condicions i, en la segona, l'existència o no de reparadors potencials. Tota condició C es pot classificar tenint en compte els dos criteris a la vegada, és a dir, serà una condició de Restricció d'Integritat o d'Actualització de Vistes, i també serà una condició de Comprovació o de Generació.

6.3.2 Precedències entre condicions

Amb la utilització del Graf de Dependències entre esdeveniments hem pogut identificar, entre altres coses, quins són els violadors i reparadors potencials de cada condició. Però en realitat, per a definir l'ordre de comprovació de les condicions ens cal conèixer quins reparadors potencials de quines condicions poden ser a la vegada violadors potencials d'altres condicions. Aquesta informació serà la que ens permetrà establir les precedències entre condicions que representarem en el Graf de Precedències.

Intuïtivament, una condició C_i precedeix una condició C_j si existeix un esdeveniment bàsic E, que depenent de si ocorre o no, la condició C_i pot esdevenir falsa i la condició C_j por ésser satisfeta. És a dir, aquest esdeveniment E és generat durant el procés de reparació de la condició C_i i pot induir una violació en la condició C_j.

Definició 6.13: Sigui C_i una condició de generació i C_j una condició. Diem que *C_i precedeix C_j degut a E* si existeix un esdeveniment bàsic E que és, a la vegada, un reparador potencial de C_i i un violador potencial de C_j.

Una precedència entre dues condicions l'escrivim de la forma següent:

$$C_i \rightarrow C_j \quad \text{degut a E}$$

on C_i, C_j es corresponen als identificadors de condicions i E és un esdeveniment bàsic. Degut al gran nombre de precedències entre condicions, aquestes també es poden representar en el format més compacte següent:

$$C_i, C_k, \ldots \rightarrow C_j, C_l, \ldots \qquad \text{degut a } E_s, E_t, \ldots$$

Exemple 6.12: Les precedències entre condicions que es poden identificar a partir del Graf de Dependències de la figura 6.3 són les següents:

C5 → C9, C10, C11, B1[1] degut a μSou

C6, C9 → C2, C3, B2 degut a ιTreb

C1, C2, C3 → C5, C7, C10, D1, B3 degut a δTreb

C1, C2, C3, C6, C9, D4 → C1, C2, C3, C4, C8, C11, D2, M1, B4 degut a μTreb

C13 → C2, C3, B5 degut a ιCont

C1, C2, C3 → C12, C14, C15, D3, B6 degut a δCont

C1, C2, C3, D2 → C1, C2, C3, C4, D4, M1, B7 degut a μCont

Es pot comprovar fàcilment que, per exemple, l'esdeveniment μSou és un reparador potencial de la condició C5 i, a la vegada, és un violador potencial de les condicions C9, C10, C11, B1.

□

Donat un conjunt de precedències, es poden donar certes situacions que cal tenir en compte. Per una banda, poden existir dependències entre condicions que formen cicles entre elles. En l'exemple 6.12, existeix un cicle entre les condicions C1→C5→C9→C1. Cal dir de totes formes, que en la majoria de casos, aquests cicles no es corresponen a cicles reals en temps d'execució. Una anàlisi més detallada d'aquests cicles es realitzarà a la secció 6.4.1.

Una altre situació que es pot detectar analitzant aquestes precedències és el fet d'existir una precedència d'una condició amb sí mateixa. Aquesta situació pot aparèixer per dues causes principalment. Per una banda, perquè en la definició d'un predicat derivat hi apareix (directament o indirectament) un predicat bàsic de forma positiva i negativa. En aquesta cas, aquest tipus de precedència té sentit per a condicions definides a partir d'aquest predicat derivat. Per altra banda, pot ésser deguda a una falta de precisió en la generació de dependències entre esdeveniments, ja que solament es considera el tipus d'esdeveniment sense tenir en compte els seus arguments. Així, per exemple, no es pot diferenciar l'esdeveniment μP(k,x,x') de l'esdeveniment μP(k,x',x). A l'exemple 6.12, és donen diferents exemples d'aquesta falta de precisió. Per exemple, la condició C1 es precedeix a sí mateixa degut a l'esdeveniment μTreb. En aquest cas, s'ha identificat aquesta precedència perquè no s'ha diferenciat entre els esdeveniments de modificació μTreb(p,c,c1) i μTreb(p,c1,c).

Un cop identificades totes les precedències entre condicions i abans de construir el Graf de Precedències s'analitzaran en detall aquest conjunt de precedències i s'aplicaran algunes simplificacions. Aquesta anàlisi té l'objectiu d'identificar aquelles precedències que no es podran satisfer realment en temps d'execució. Per exemple, amb aquesta anàlisi s'arranjarà la falta de precisió comentada en el paràgraf anterior.

---

[1] La condició B1 es correspon a la condició ←μSou(p,c,s,c1,s1). De forma equivalent la resta de condicions Bi es corresponen a condicions ←Ev, on Ev és l'esdeveniment causant de la precedència.

6.3.3 Optimitzacions

Cada una de les precedències ($C_i \rightarrow C_j$ degut a E) que han estat identificades en la secció anterior estableixen que al reparar una condició C_i es pot induir una violació d'una altra condició C_j. Si s'analitzen amb més detall aquestes precedències, es pot detectar que algunes d'elles no es podran satisfer realment en temps d'execució degut a que els requeriments necessaris per a reparar la condició C_i i violar la condició C_j són incompatibles i no es poden satisfer a la vegada. És a dir, que en temps d'execució, al reparar la condició C_i amb l'esdeveniment E no és possible induir una violació de la condició C_j.

En aquesta secció proposem realitzar una anàlisi individualitzada de cada una d'aquestes precedències ($C_i \rightarrow C_j$ degut a E) identificar aquelles precedències que en temps d'execució, no es podran satisfer i eliminar-les del conjunt final de precedències a representar en el Graf de Precedències. Aquesta anàlisi sols té en compte la informació sintàctica de cada una de les condicions i la definició dels esdeveniments involucrats en cada condició. Com més detallada es realitzi aquesta anàlisi, més eficient aconseguirem que sigui el procés de manteniment de la consistència, ja que totes aquestes precedències seran eliminades i s'obtindrà un Graf de Precedències simplificat i optimitzat.

En concret, per a cada precedència ($C_i \rightarrow C_j$ degut a E) s'analitzen els requeriments (esdeveniments i fets de la base de dades) que s'han de satisfer per a violar la condició C_i. Un cop determinats aquests, es determina quin ha de ser exactament l'esdeveniment E. A continuació, es determinen quins són els requeriments (esdeveniments i fets de la base de dades) que s'hauran de satisfer per a que l'esdeveniment E violi realment la condició C_j. Si es detecta alguna contradicció en el conjunt d'aquests requeriments llavors es pot eliminar la precedència. En cas contrari, aquesta precedència es representarà en el Graf de Precedències.

En el cas que en alguna condició hi apareguin referències a predicats i/o esdeveniments derivats, cal tenir en compte totes les regles de derivació d'aquests predicats i totes les regles d'esdeveniment associades. En aquests casos, una precedència solament es pot eliminar si i només si no és assolible per totes i cada una d'aquestes regles. En el cas de que per alguna d'aquestes regles sigui possible reparar C_i i violar C_j mitjançant l'esdeveniment E, aleshores la precedència ($C_i \rightarrow C_j$ degut a E) ha d'incloure's al Graf de Precedències.

Dues són les situacions que es pretenen detectar per tal d'optimitzar el conjunt de precedències a representar en el Graf de Precedències: l'ocurrència de dos esdeveniments mútuament exclusius i la necessitat de satisfer dos requeriments contradictoris.

Optimització I: Ocurrència de dos esdeveniments mútuament exclusius

Com ja s'ha definit al capítol 4 d'aquest document, els esdeveniments que pertanyen a una transacció T no poden ser mútuament exclusius. Per tant, si una precedència entre condicions requereix que dos esdeveniments mútuament exclusius hagin de pertànyer a la mateixa

transacció T, llavors aquesta precedència pot ser eliminada, ja que en temps d'execució aquests dos esdeveniments no poden ocórrer simultàniament.

En l'exemple següent es mostra una precedència de l'exemple 6.12 que pot ser eliminada per requerir l'execució de dos esdeveniments mútuament exclusius.

Exemple 6.13: Consideri's la precedència obtinguda a l'exemple 6.12 "C5 → C11 degut a μSou" i la definició de les condicions involucrades:

(C5) $\leftarrow Sou(\underline{p},c,s) \wedge \neg\delta Sou(\underline{p},c,s) \wedge \neg Aux6(\underline{p},c,s) \wedge Treb(\underline{p},c) \wedge \delta Treb(\underline{p},c)$

(C11) $\leftarrow Sou(\underline{p},c1,s1) \wedge \mu Sou(\underline{p},c1,s1,c,s) \wedge c1 \neq c \wedge s1 \neq s \wedge Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c2)$

(A6) $Aux6(p,c,s) \leftarrow \mu Sou(\underline{p},c,s,c1,s1)$

La condició C5 esdevindrà violada sempre que es compleixin els requeriments següents:

- $\delta Treb(\underline{p},c) \in T$
- Els fets $Treb(\underline{p},c)$ i $Sou(\underline{p},c,s)$ són certs a la base de dades extensional
- $\delta Sou(\underline{p},c,s) \notin T$, $\mu Sou(\underline{p},c,s,c1,s1) \notin T$ amb $c \neq c1$

Un reparador potencial de la condició C5 és $E = \mu Sou(\underline{p},c,s,c1,s1)$ sempre que $c \neq c1$.

L'esdeveniment E pot violar la condició C11 en el cas que (entre altres requeriments):

- El fet $Sou(\underline{p},c,s)$ és cert a la base de dades extensional i
- $\mu Treb(\underline{p},c1,c2) \in T$

Observi's que en aquest cas, l'esdeveniment $\mu Treb(\underline{p},c1,c2)$ és mútuament exclusiu amb l'esdeveniment $\delta Treb(\underline{p},c)$ necessari per violar C5. Concretament, no satisfan la definició d'exclusivitat següent: $\forall k,x,x'\ (\mu P(\underline{k}, x, x') \rightarrow \neg\ \exists\ y\ \delta P(\underline{k}, y))$. Per tant, els dos esdeveniments no poden pertànyer a la mateixa transacció T i aquesta precedència pot ser eliminada ja que no es podrà satisfer en temps d'execució.

□

Optimització II: Dos requeriments contradictoris

Una situació que permet eliminar una precedència entre dues condicions apareix quan per a satisfer una precedència cal assumir dos requeriments contradictoris o oposats. Aquesta situació es pot donar perquè es requereix que un fet sigui cert i fals en el mateix estat de la base de dades, o perquè s'ha d'assumir que un esdeveniment concret ha d'ocórrer i s'ha d'evitar durant la transició entre els mateixos estats de la base de dades.

Exemples concrets d'aquestes situacions es mostren a continuació.

Exemple 6.14: Considerem la precedència identificada a l'exemple 6.12 "C2 → C4 degut a μCont" i la definició de les condicions involucrades:

(C2) $\leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Edat(\underline{p}) \wedge \neg\iota Edat(\underline{p})$

(C4) $\leftarrow Emp(\underline{p},c) \wedge \mu Emp(\underline{p},c,c1) \wedge c1 \neq c \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(A11) $Aux11(\underline{p}) \leftarrow Emp(\underline{p},c)$

Alguns requeriments que s'han de satisfer perquè la condició C2 esdevingui violada són:

- El fet bàsic Edat(p) ha de ser fals en la base de dades extensional
- $\neg\exists c$ tal que Emp(p,c) sigui cert a la base de dades

Per tal que la condició C4 pugui ser reparada, cal que se satisfacin (entre d'altres) els requeriments següents:

- El fet bàsic Edat(p) ha de ser cert a la base de dades extensional
- El fet derivat Emp(p,c) ha de ser cert a la base de dades

Observi's que aquests requeriments sobre el contingut de la base de dades són contradictoris, i per tant, quan la condició C2 sigui violada per alguna transacció T no serà possible violar la condició C4 al mateix temps, independentment de la reparació aplicada. Per tant, tota precedència entre les condicions C2 i C4 pot ser eliminada del conjunt de precedències a representar en el Graf de Precedències.

□

Exemple 6.15: Consideri's en aquest exemple la precedència "D4 $\rightarrow$ D2 degut a μTreb" i la definició de les condicions involucrades:

(D2) $\leftarrow Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c1) \wedge c1 \neq c \wedge Cont(\underline{p},c) \wedge \neg\mu Cont(\underline{p},c,c1)$

(D4) $\leftarrow Treb(\underline{p},c) \wedge Cont(\underline{p},c) \wedge \mu Cont(\underline{p},c,c1) \wedge c1 \neq c \wedge \neg\mu Treb(\underline{p},c,c1)$

Els requeriments que s'han de satisfer perquè la condició D4 esdevingui violada són:

- El fets bàsics Treb(p,c) i Cont(p,c) han de ser certs
- $\mu Cont(\underline{p},c,c1) \in T$ i $\mu Treb(\underline{p},c,c1) \notin T$ amb $c1 \neq c$

L'esdeveniment E = μTreb(p,c,c1) és el reparador de la condició D4.

Per a que la condició D2 esdevingui violada per l'esdeveniment E, cal que:

- El fets bàsics Treb(p,c) i Cont(p,c) siguin certs a la base de dades extensional
- $\mu Cont(\underline{p},c,c1) \notin T$

Observi's que en aquest cas, l'esdeveniment μCont(p,c,c1) ha d'ocórrer per a violar la condició D4, però no pot ocórrer per a violar la condició D2. Per tant, aquesta precedència entre les condicions D4 i D2 pot ser eliminada del conjunt de precedències perquè amb la reparació de D4 amb l'esdeveniment μTreb mai es podrà violar la condició D2.

□

Aquestes dues optimitzacions sintàctiques (I i II) són les que proposem aplicar al conjunt de precedències identificades abans de construir el Graf de Precedències. Amb aquestes optimitzacions, es poden eliminar gran part de precedències i simplificar considerablement el Graf de Precedències. En molts casos, aquestes optimitzacions permeten eliminar cicles que existeixen entre precedències, i és per aquesta raó, per la que és especialment beneficiosa l'aplicació d'aquestes optimitzacions.

Exemple 6.16: Un cop aplicades les optimitzacions I i II a les precedències de l'exemple 6.12, hem obtingut el conjunt final de precedències següent:

C5 → C9, C10, B1	degut a μSou
C6, C9 → C2, C3, B2	degut a ιTreb
C1, C2, C3 → C5, C7, C10, D1, B3	degut a δTreb
C1, C2, C3, C6, C9 → C8, C11, B4	degut a μTreb
C1, C6, C9 → C4, D2, M1	degut a μTreb
C6, C9 → C1, C2, C3	degut a μTreb
D4 → C1, C4, C8, C11, M1, B4	degut a μTreb
C13 → C2, C3, B5	degut a ιCont
C1, C2, C3 → C12, C14, C15, D3, B6	degut a δCont
C1 → C4, D4, M1, B7	degut a μCont
D2 → C1, C4, M1, B7	degut a μCont

□

L'optimització que proposem a continuació, no es correspon a una optimització sintàctica de cada una de les precedències, com en aquests casos, sinó que és una optimització que està més relacionada amb l'estructura de les condicions B_i i en la forma com hauran de ser considerades en temps d'execució.

Optimització III: Eliminació dels nodes B_i del Graf de Precedències

Cada node etiquetat amb una etiqueta B_i té associada una condició d'actualització de vistes amb el format següent:

$(B_i) \quad \leftarrow Ev$

En el cos de la condició sols hi ha un únic literal i aquest es correspon a un esdeveniment bàsic (Ev). Com ja hem comentat anteriorment, aquestes condicions són condicions de comprovació, ja que no tenen associat cap reparador potencial i l'únic violador potencial que tenen és precisament l'esdeveniment Ev. Aquestes condicions, al no tenir cap reparador potencial, sols poden estar involucrades en alguna precedència com a condició violada (C_j).

La principal particularitat d'aquestes condicions, és la simplicitat del tractament que cal aplicar, en temps d'execució, per a comprovar si estan o no violades. Sols cal comprovar si

l'esdeveniment bàsic ha ocorregut o no, és a dir, si Ev∈ T. En cas afirmatiu, sempre es rebutja la transacció T, ja que no poden ser reparades. A més a més, aquestes condicions són (com veurem a la secció 6.4) les que es proposaran comprovar en primer lloc ja que, quan més aviat es detecti que estan violades, més aviat podrem rebutjar la transacció T, i així evitar tenir de fer comprovacions i reparacions d'altres condicions que es rebutjarien igualment.

Tenint en compte aquestes consideracions, la tercera optimització que proposem consisteix en eliminar totes les precedències en que participen aquestes condicions, de forma que no representarem cap node etiquetat com B_j en el Graf de Precedències. En el seu lloc, proposem acumular totes les condicions d'aquest tipus, a mida que es vagin generant, en un conjunt auxiliar de condicions de comprovació que anomenarem Cc, i que es comprovaran cada cop que s'afegeixi un nou esdeveniment al conjunt de traducció T. Observi's que aquest conjunt és equivalent al conjunt de condició C, però en aquest cas, les condicions que conté són únicament del tipus ← Ev.

Definició 6.14: Anomenem *Conjunt de Condicions de Comprovació Cc* al conjunt de condicions d'actualització de vistes que tenen el format ← Ev, on Ev és un esdeveniment bàsic.

Aquesta optimització reduirà considerablement la complexitat del Graf de Precedències, ja que elimina un gran nombre de precedències i a la vegada un gran nombre de nodes d'aquest Graf, tants com esdeveniments bàsics, es poden definir a partir dels predicats bàsics de la base de dades. Per altra banda, al ser condicions de comprovació i al comprovar-les cada cop que es repara alguna altra condició (es canvia el conjunt T), permetrà rebutjar el més aviat possible la transacció T, reduint treball innecessari. En canvi, a l'acumular aquestes condicions en un conjunt auxiliar Cc i no tenir explícites les precedències en que estan involucrades, no permetrà reduir el nombre de vegades que cal recomprovar-les, ja que es comprovaran tants cops com reparacions es produeixin. Però cal tenir en compte també, que el cost que comporta la comprovació de cada una d'aquestes condicions és mínim.

Exemple 6.17: En aquest exemple es mostra el conjunt final de les precedències que s'han obtingut un cop aplicades les tres optimitzacions proposades en aquesta secció:

C5 → C9, C10	degut a μSou
C6, C9 → C2, C3	degut a ιTreb
C1, C2, C3 → C5, C7, C10, D1	degut a δTreb
C1, C2, C3, C6, C9 → C8, C11	degut a μTreb
C1, C6, C9 → C4, D2, M1	degut a μTreb
C6, C9 → C1, C2, C3	degut a μTreb
D4 → C1, C4, C8, C11, M1	degut a μTreb
C13 → C2, C3	degut a ιCont
C1, C2, C3 → C12, C14, C15, D3	degut a δCont

C1 → C4, D4, M1 degut a μCont

D2 → C1, C4, M1 degut a μCont

□

6.3.4 Graf de Precedències

Un cop identificades i optimitzades totes les precedències entre condicions, podem construir el Graf de Precedències, el qual estableix totes les interrelacions entre reparadors potencials i violadors potencials de condicions.

En aquest graf podem trobar definit l'ordre (o possibles ordres) en que cal comprovar les diferents condicions que ens asseguren que una transacció T és consistent respecte a les restriccions d'integritat i a la petició inicial d'actualització. Aquest ordre de comprovació de condicions ens permet, per una banda, reduir el nombre de vegades que cada condició ha de ser comprovada. D'aquesta forma, en temps d'execució, es pot incrementar considerablement l'eficiència del procés de manteniment de la consistència. Per l'altra banda, aquest ordre ens permet assegurar que no es repararà cap condició C fins que totes les condicions, les reparacions de les quals poden violar C, hagin estat satisfactòriament reparades.

Definició 6.15: Donat un conjunt de condicions, un Graf de Precedències GP és una terna GP=<N, G, A> on N es un conjunt finit de nodes, G és un conjunt de subgrafs de precedències i $A \subseteq (N \times N) \cup (G \times N)$ és el conjunt de arestes dirigides i etiquetades. Cada node $n \in N$ està etiquetat amb l'identificador d'una condició, cada subgraf $g \in G$ està etiquetat amb un identificador ad hoc (g_i) i cada aresta $a \in A$ està etiquetada amb un esdeveniment bàsic. Existeixen dos tipus d'arestes:

- una aresta a=(n,n') entre dos nodes n i n', etiquetada amb l'esdeveniment Ev, indica l'existència d'una precedència des de n fins a n' deguda a l'esdeveniment Ev.
- una aresta a=(g,n') entre un subgraf g i un node n', etiquetada amb l'esdeveniment Ev, indica l'existència d'una precedència des d'algun node del subgraf g fins a n' deguda a l'esdeveniment Ev.

Els subgrafs que apareixen en un Graf de Precedències són un cas especial de node. Un subconjunt de nodes del graf de precedència que formen un cicle, s'agrupen en un node subgraf el qual s'etiqueta amb un identificador ad hoc g_i. Aquests subgrafs tenen la mateixa estructura que un graf de precedències. Els nodes subgraf s'han introduït per a representar el fet de que la reparació d'una condició C pot violar altres condicions que al ser reparades, poden tornar a violar la primera condició C. En aquests casos, no es pot assegurar que la condició C estigui satisfactòriament reparada fins que totes les condicions del cic. hagin estat satisfactòriament reparades. És precisament per aquesta raó, que totes les precedències entre un node de dins d'un subgraf i un node extern es representen com a arestes entre el subgraf i el node extern.

Exemple 6.18: Tenint en compte les precedències obtingudes a l'exemple 6.17, un cop aplicades les tres optimitzacions anteriors, les podem representar en el Graf de Precedències. La figura 6.4 mostra el Graf de Precedències resultant per aquest exemple. □

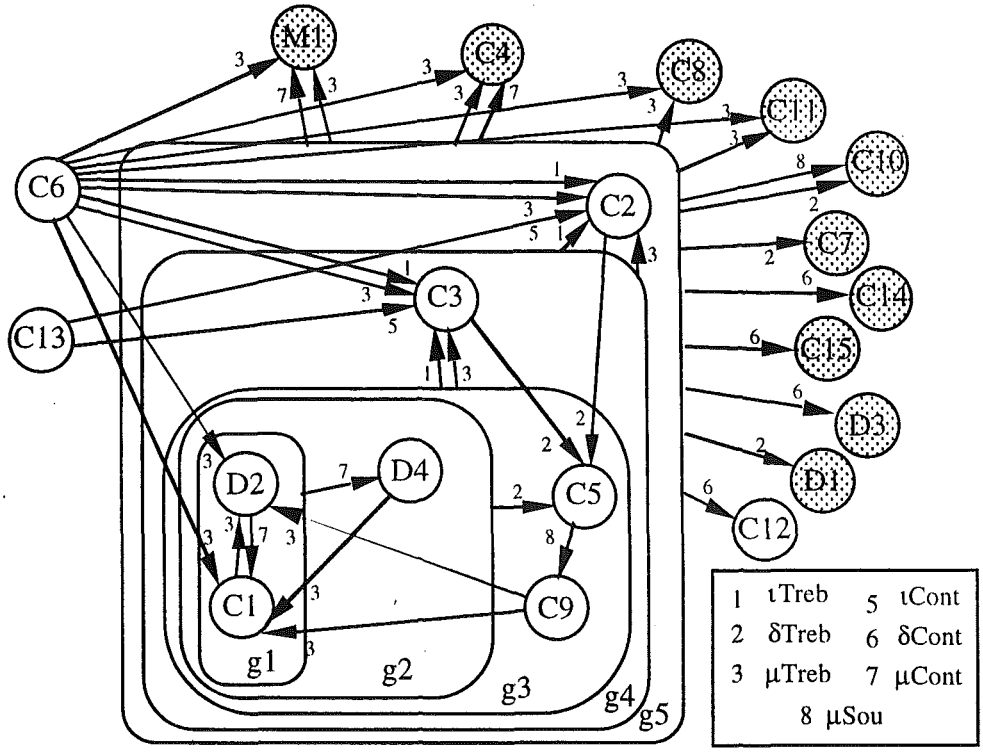


Fig. 6.4. Graf de Precedències entre condicions

En un Graf de Precedències de la figura 6.4 es poden identificar diferents situacions que cal comentar:

- un node representat amb fons blanc: indica que la condició associada a aquest node és una condició de generació
- un node representat amb fons gris: indica que la condició associada a aquest node és una condició de comprovació
- un node sense arestes de sortida: indica que les reparacions de la condició associada a aquest node no poden violar cap altre condició del graf
- un node sense arestes d'arribada: indica que la condició associada a aquest node no pot ser violada per la reparació de cap altre condició del graf
- un subgraf amb alguna aresta de sortida cap a un node: indica que la reparació d'alguna condició associada a un node del subgraf (cicle) pot violar la condició associada al node on arriba l'aresta
- no existeixen arestes d'arribada a un subgraf, ja que les arestes indiquen sempre quina condició concreta pot ser violada per la reparació d'una altre condició

En el Graf de Precedències de la figura 6.4 es pot observar que apareixen diversos cicles entre condicions. Generalment no apareixen molts cicles entre nodes, però és una situació que pot aparèixer en determinats exemples. En aquest cas, apareixen degut a la complexitat de l'exemple escollit.

Exemple 6.19: A partir del Graf de Precedències de la figura 6.4, es poden observar algunes d'aquestes situacions:

- la condició del node C1 és una condició de generació
- la condició del node C11 és una condició de comprovació
- en aquest cas, sols existeix una condició de generació (C12) que al ser reparada no pot violar cap altre condició del graf
- les condicions C6 i C13 no poden ser violades per la reparació d'una altre condició
- la condició D4 pot ser violada per la reparació d'alguna condició associada a algun node del subgraf g1. El que no es pot observar en el graf és quina de les dues condicions (C1 o D2) és la causant de la violació
- en canvi sí que es representa en el graf que la reparació de la condició D4 pot violar la condició del node C1.

En la figura 6.4, existeixen cinc subgrafs i en cada un d'ells hi ha associat un únic cicle entre nodes. En aquest cas, com que els diferents cicles que es poden identificar entre les precedències de l'exemple 6.17 tenen nodes en comú, llavors en el Graf de Precedències hi apareixen subgrafs inclosos en altres subgrafs. Una forma alternativa de representar aquests cicles, seria en un únic subgraf que inclogués els nodes C1, C2, C3, C5, C9, D2 i D4. Però, en aquest cas, no hi hauria associat un únic cicle per subgraf.

□

És especialment important tenir en compte que en el Graf de Precedències sols es representen els violadors (reparadors) potencials d'una condició si i sols si aquests són a la vegada reparadors (violadors) potencials d'altres condicions representades en el graf. Així doncs, aquest graf no proporciona informació de quins són tots els violadors i reparadors potencials d'una condició. Aquesta informació es pot obtenir del Graf de Dependències entre esdeveniments definit a la secció 6.3.1.

Per altra banda, cal comentar que l'existència de cicles en el Graf de Precedències pot donar la idea de que el procés de manteniment de la consistència és infinit, en el sentit que no acaba. Respecte a aquest punt cal esmentar que totes les precedències e e condicions del Graf de Precedències són potencials, i que en temps d'execució, aquests cicles no tenen perquè correspondre's a execucions infinites. A la secció 6.4, en la que es presenta el mecanisme de manteniment de la consistència utilitzant el Graf de Precedències s'analitza en detall perquè aquests cicles generalment no es corresponen en realitat a cicles infinits.

6.3.5 Alternatives al disseny del Graf de Precedències

En aquest apartat, s'analitzen dues alternatives al disseny del Graf de Precedències. Per una banda, s'analitzen les implicacions que tindria el considerar un Graf de Precedències on els nodes, en lloc de tenir associades condicions de restriccions d'integritat, hi haguessin associades directament els esdeveniments d'inserció dels predicats d'inconsistència de cada restricció d'integritat. Per altra banda, s'analitza el cas en que les arestes entre nodes puguin estar etiquetades tant amb esdeveniments bàsics com amb esdeveniments derivats.

Alternativa I: Restriccions d'integritat en lloc de condicions als nodes

Tenint en compte altres propostes per al manteniment de restriccions d'integritat, com les presentades a [Ger94, CFPT94], ens hem plantejat la possibilitat de definir un Graf de Precedències alternatiu en el que els nodes tenen associats l'esdeveniment d'inserció del predicat de inconsistència associat a cada restricció d'integritat en lloc de les condicions associades a aquestes restriccions. Les arestes segueixen etiquetant-se amb esdeveniments bàsics.

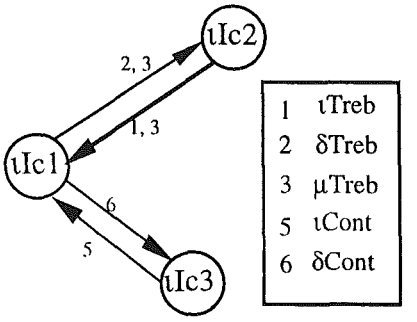
Una primera observació que cal fer a aquesta alternativa és que la informació associada als nodes del graf no seria la mateixa. Els nodes que tenen associades restriccions d'integritat contindrien l'esdeveniment de inconsistència (ιIc_i). La resta de nodes contindrien les condicions d'actualització de vistes del conjunt C. Aquest fet fa que en temps d'execució calgui definir dos tractaments diferents per a tractar aquests dos tipus de nodes.

Aquesta alternativa podria ser més adequada per als casos en que no es considerés el problema de l'actualització de vistes, com en les propostes anteriors [Ger94, CFPT94], o en els casos en que les condicions del conjunt C obtingudes durant un procés d'actualització de vistes es mantinguessin de forma independent al procés de manteniment de la consistència de les restriccions d'integritat. En aquest segon cas, no es podríem representar les precedències entre els dos tipus de condicions i es perdria eficiència en el procés de manteniment de la consistència respecte a l'alternativa de la secció 6.3.4.

En l'exemple següent, es mostra el Graf de Precedències que s'obtindria associat a l'exemple 3.1 tenint en compte aquesta alternativa.

Exemple 6.20: A la figura 6.5 es mostra el Graf de Precedències associat a aquesta primera alternativa I. Observi's que cada node representa l'esdeveniment d'inserció d'un predicat d'inconsistència en lloc de les diferents formes com es pot induir aquest esdeveniment (condicions de restriccions d'integritat). A més a més, les condicions associades al procés d'actualització de vistes no s'han representat en el graf.

□



6.5 Graf de Precedències de la alternativa I

Comparant aquest Graf de Precedències amb l'obtingut a la secció anterior (figura 6.4) podem observar algunes diferències. El Graf de Precedències d'aquesta alternativa és més simple perquè el nombre de nodes i arestes és inferior respecte al de la figura 6.4. Però, cal dir de totes maneres, que el nombre de cicles entre nodes és superior respecte a la proposta de graf de la secció 6.3.4. Aquest fet és especialment inconvenient pel nostre propòsit, ja que com més cicles contingui el Graf de Precedències, menys útil és l'ús del Graf de Precedències per a determinar un ordre de tractament de les restriccions d'integritat.

Per altra banda, cal observar que en aquesta alternativa s'ha perdut precisió en la informació representada en el Graf de Precedències. En aquesta alternativa, un node representa una restricció d'integritat, mentre que en la proposta de la secció 6.3.4, cada node representa una forma diferent de violar una restricció d'integritat. Pel propòsit pel que definim el Graf de Precedències, que és la millora de l'eficiència en la detecció i reparació de violacions de restriccions d'integritat, és més adequada una associació dels nodes amb les condicions de les restriccions d'integritat, que una associació amb la pròpia definició de les restriccions, ja que en qualsevol cas, el nostre mètode comprova i repara les regles d'esdeveniment ιIc_i. La major precisió de la informació representada, ens permet ser més precisos i eficients en el tractament dels nodes del Graf.

Així doncs, tenint en compte aquestes limitacions, creiem que pel nostre propòsit és més adequada la definició de Graf de Precedències proposada a la secció 6.3.4. De totes formes, no descartem que aquesta alternativa pugui ser especialment útil per a altres problemes o enfocaments.

Alternativa II: Arestes etiquetades amb esdeveniments derivats

En la definició del Graf de Precedències proposada a la secció 6.3.4, les arestes entre nodes estan sempre etiquetades amb un esdeveniment bàsic. Aquest esdeveniment es correspon al violador i reparador potencial de les condicions associades als nodes involucrades en cada precedència.

En el cas de tenir condicions definides pe. sdeveniments derivats, el definir les precedències entre aquestes condicions mitjançant esdeveniments bàsics pot donar lloc a una pèrdua de la intuïció del concepte de precedència. Per a interpretar una precedència caldrà

analitzar de quina manera l'esdeveniment bàsic que determina la precedència defineix algun esdeveniment derivat dels que apareixen a cada condició. Aquesta anàlisi pot ser especialment difícil de fer com més nivells de derivació i més complexes siguin la definició de les vistes i les regles d'esdeveniments de la base de dades.

Per tal d'evitar aquesta pèrdua de la intuïció o poder expressiu en la definició de les precedències entre condicions, hem plantejat una alternativa a la definició de les precedències. Aquesta alternativa consisteix a permetre que les precedències entre condicions es defineixin en termes d'esdeveniments derivats i d'esdeveniments bàsics, depenent de la pròpia definició de les condicions involucrades en la precedència. Una dificultat inherent a aquesta major flexibilitat rau en la determinació, per a cada parell de condicions, de quin esdeveniment derivat o bàsic escollir per a definir la precedència. El Graf de Precedències, un cop identificades totes les precedències entre condicions, es representarà de la mateixa manera com s'ha fet en la secció 6.3.4.

Aquest canvi en el nivell de definició de les arestes del graf sols ha de comportar, en temps de definició, una representació més intuïtiva de les precedències entre condicions, i en temps d'execució, comportarà aplicar petites modificacions a l'algorisme encarregat del tractament de les condicions del mòdul de manteniment de la consistència. Però abans d'analitzar els avantatges i limitacions d'aquesta alternativa, hem d'ampliar algunes de les definicions de la secció 6.3.1 i 6.3.2 per a considerar correctament aquesta alternativa.

El concepte de condició, i els criteris utilitzats per a classificar-les com a condicions de restriccions d'integritat o d'actualització de vistes, i condicions de comprovació o de generació, segueixen essent els mateixos per a aquesta alternativa.

L'esdeveniment bàsic o derivat que ha de servir com a punt de referència per a identificar i definir una precedència entre dues condicions és el que anomenarem *esdeveniment de colapse*.

Definició 6.16: Siguin C_i i C_j dues condicions diferents. Un esdeveniment E (bàsic o derivat) és un *esdeveniment de colapse* si es compleix alguna de les següents situacions:

- si E apareix a la definició de C_i i de C_j
- si E apareix a la definició de C_i i defineix un esdeveniment derivat E' que apareix a la definició de C_j però no a la definició de C_i
- si E defineix dos esdeveniments derivats E' i E", tal que E' apareix a la definició de C_i, i E" apareix a la definició de C_j però no a la definició de C_i

Exemple 6.21: Donades les condicions C2, C3, C4, C10 i C11 següents:

(C2) $\leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Edat(\underline{p}) \wedge \neg\iota Edat(\underline{p})$

(C3) $\leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(C4) $\leftarrow Emp(\underline{p},c) \wedge \mu Emp(\underline{p},c,c1) \wedge c1 \neq c \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(C10) $\leftarrow Sou(\underline{p},c1,s1) \wedge \mu Sou(\underline{p},c1,s1,c,s) \wedge c1 \neq c \wedge s1 \neq s \wedge Treb(\underline{p},c) \wedge \delta Treb(\underline{p},c)$

(C11) $\leftarrow Sou(\underline{p},c1,s1) \wedge \mu Sou(\underline{p},c1,s1,c,s) \wedge c1 \neq c \wedge s1 \neq s \wedge Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c2)$

(A11) $Aux11(\underline{p}) \leftarrow Emp(\underline{p},c)$

Un esdeveniment de colapse entre les condicions C10 i C11 és l'esdeveniment $\mu Sou(\underline{p},c1,s1,c,s)$ perquè apareix explícitament en la definició de les dues condicions. Per la mateixa raó, l'esdeveniment derivat $\iota Emp(\underline{p},c)$ és un esdeveniment de colapse entre les condicions C2 i C3. Per altra banda, l'esdeveniment bàsic $\delta Treb(\underline{p},c)$ és un esdeveniment de colapse entre les condicions C2 i C10 perquè apareix a la definició de C10 i defineix l'esdeveniment $\iota Emp(\underline{p},c)$ que apareix a C2 (i no apareix a C10). Finalment, podem considerar que l'esdeveniment $\mu Treb(\underline{p},c,c1)$ és un esdeveniment de colapse entre les condicions C2 i C4 ja que defineix els esdeveniments derivats $\iota Emp(\underline{p},c)$ i $\mu Emp(\underline{p},c,c1)$ que apareixen únicament a les respectives condicions C2 i C4.

□

La definició de Graf de Dependències entre esdeveniments de la secció 6.3.1 segueix essent vàlida per a aquesta alternativa. Però, abans de definir el concepte de precedència entre dues condicions degut a un esdeveniment de colapse, cal adaptar els conceptes de violador i reparador potencial, i introduir el concepte de falsejador potencial.

Els conceptes de violador i reparador potencials d'una condició segueixen essent vàlids, però, en aquesta alternativa, els generalitzarem per a esdeveniments derivats. Així doncs, parlarem de violadors i reparadors potencials d'una condició o d'un esdeveniment derivat. Per altra banda, en aquesta alternativa, un violador potencial tant pot ser un esdeveniment bàsic com derivat, ja que la seva ocurrència pot fer esdevenir certa una condició o induir un esdeveniment derivat. En canvi, ens interessarà distingir entre un reparador potencial i un falsejador potencial. Els dos fan referència a un esdeveniment que a l'ocórrer pot fer falsa una condició o esdeveniment derivat, però distingim quan aquest és un esdeveniment bàsic (reparador potencial) de quan és un esdeveniment derivat (falsejador potencial).

Definició 6.11bis: Sigui C una condició o un esdeveniment derivat i E un esdeveniment (bàsic o derivat).

a) E és un *violador potencial* de C si C depèn de forma parell de E.

b) E és un *reparador potencial* de C si E és un esdeveniment bàsic i C depèn de forma senar de E.

c) E és un *falsejador potencial* de C si E és un esdeveniment derivat i C depèn de forma senar de E.

Exemple 6.22: Tenint en compte la condició C1 i el Graf de Dependències entre esdeveniments de la figura 6.3, podem dir que aquesta condició té dos falsejadors potencials

δEmp(p,c) i μEmp(p,c1,c). Els violador i reparadors potencials d'aquesta condició són els mateixos que tenia a l'alternativa de la secció 6.3.1.

□

En aquesta alternativa, les precedències entre condicions estaran determinades per esdeveniments de colapse.

Definició 6.13bis: Sigui C_i una condició de generació i C_j una condició. Diem que *C_i precedeix C_j degut a E* si:

a) existeix un esdeveniment de colapse (bàsic o derivat) E tal que és un reparador o falsejador potencial de C_i i és al mateix temps un violador potencial de C_j, o

b) existeix un esdeveniment de colapse derivat E tal que C_i depèn de forma parell de E i C_j en depèn de forma senar.

Exemple 6.23: Consideri's les condicions C1 i C4. En aquest cas, les precedències que existien entre aquestes condicions a l'exemple 6.17 eren les següents: C1 → C4 degut a μTreb i C1 → C4 degut a μCont. En aquesta alternativa, al considerar l'esdeveniment derivat μEmp com a potencial falsejador de C4 i potencial violador de C1, las anteriors precedències quedaran substituïdes per la nova precedència C1 → C4 degut a μEmp.

□

Un cop obtingudes totes les precedències entre condicions, i un cop optimitzades utilitzant les optimitzacions definides a la secció 6.3.3, ja podem construir el Graf de Precedències de la mateixa manera que en la secció 6.3.4.

El Graf de Precedències obtingut considerant aquesta alternativa és molt similar al graf de Precedències obtingut en la secció 6.3.4. En nombre de nodes és el mateix i solament han canviat algunes arestes entre nodes. Bàsicament, les principals diferències són dues:

1. algunes arestes han canviat la seva etiqueta referenciant a un esdeveniment derivat en lloc d'un esdeveniment bàsic
2. quan entre dos nodes hi ha més d'una aresta, algunes d'aquestes poden substituir-se per una sola aresta etiquetada per un esdeveniment derivat.

Es pot comprovar, que encara que s'ha millorat com transmetre d'una forma més simple la idea intuïtiva d'una precedència entre dues condicions definides per esdeveniments derivats, amb aquesta alternativa, s'ha perdut precisió global a nivell de tot el Graf de Precedències. Amb aquesta alternativa, ja no està representada de forma tant explícita la relació entre l'esdeveniment reparador d'una condició i el potencial violador d'altres condicions. En aquest cas, l'esdeveniment reparador s'ha de calcular expressament, a partir del falsejador potencial que etiqueta l'aresta.

Cada cop que s'hagi reparat una condició, per a conèixer quines són les condicions que poden esdevenir violades, caldrà calcular quins esdeveniments derivats poden ser induïts a

partir dels esdeveniments bàsics responsables de la reparació. És potser aquest, el gran inconvenient d'aquesta alternativa respecte a la proposta de la secció 6.3.4, ja que aquest procés haurà de realitzar-se després de cada reparació i comportarà un cost addicional respecte a la proposta de la secció 6.3.4, on aquesta informació està explícita en el Graf de Precedències.

6.4 Mòdul de Manteniment de la Consistència

El procediment de manteniment de la consistència consisteix en, donada una transacció T i un Graf de Precedències, obtenir totes les solucions mínimes S_i tal que $T \subseteq S_i$ i que no violen cap de les condicions del Graf de Precedències. A més a més, amb aquest procés es pretén que cada condició sigui comprovada i reparada el menor nombre de vegades.

Per a iniciar el procés de comprovació de la consistència cal tenir definit el Graf de Precedències generat en temps de compilació i una traducció TC composada per la transacció T i el conjunt de condicions C. En el cas que prèviament a l'execució del mòdul de manteniment de la consistència, s'hagi executat el mòdul d'actualització de vistes, la traducció TC serà una de les obtingudes per aquest procés. En cas contrari, la transacció T contindrà els esdeveniments bàsics positius de la petició d'actualització U i el conjunt C contindrà les condicions associades als esdeveniments negats de la petició U.

El resultat de l'execució d'aquest mòdul és el conjunt de totes les solucions mínimes S_i ($T \subseteq S_i$) tal que no violen cap de les condicions del graf. Les actualitzacions que pertanyen al conjunt S_i-T es corresponen a les actualitzacions que ha calgut afegir a la transacció T per tal de reparar les condicions del Graf de Precedències, violades per T.

Així doncs, el mecanisme per a mantenir les condicions representades en el Graf de Precedències consisteix en determinar quines de les condicions del graf cal comprovar i assegurar-se que cap d'elles és violada per la transacció T seguint l'ordre determinat pel propi Graf de Precedències. Aquest ordre de tractament de les condicions ha d'assegurar que al final del procés de manteniment de la consistència, cap condició es violi i que tota condició s'hagi comprovat el menor nombre de vegades.

Quan alguna de les condicions sigui violada per T, es repararà i es continuarà amb la següent condició. Quan per a reparar una condició calgui actualitzar un fet derivat, la reparació serà obtinguda pel mòdul d'Actualització de Vistes. Un cop comprovades i reparades totes les condicions, la transacció finalment obtinguda contindrà totes les actualitzacions necessàries per a satisfer la petició d'actualització U i les restri( )ns d'integritat. Obtindrem tantes solucions S com formes diferents hi hagi per satisfer la petició d'actualització inicial sense violar cap condició.

Abans de presentar l'algorisme per al manteniment de la consistència d'una transacció T utilitzant el Graf de Dependències, comentarem certes consideracions relatives al mecanisme

que definirem i com interpretar i manegar la informació representada en el Graf de Precedències.

6.4.1 Consideracions prèvies

Aquestes consideracions cal tenir-les en compte per tal de comprendre amb més detall l'algorisme proposat pel manteniment de la consistència utilitzant un Graf de Precedències.

Nodes Actius i Graf de Precedències Actiu

En primer lloc, cal recordar que en els nodes del Graf de Precedències hi ha dos tipus de condicions representades: les condicions de restriccions d'integritat i les condicions d'actualització de vistes. Les primeres són condicions que asseguren la no violació de cap restricció d'integritat i, per tant, tota transacció T que es vulgui aplicar sobre la base de dades, les ha de satisfer. Per altra banda, el segon tipus de condicions són generades per un procés d'actualització de vistes i sols han de ser comprovades per les transaccions generades en aquest mateix procés de traducció.

Així doncs, podem observar que no totes les transaccions han de satisfer totes les condicions representades en el Graf de Precedències, per tant, no tots els nodes del Graf han de ser considerats per tota transacció T. Els nodes que tenen associades condicions de restriccions d'integritat han de ser considerats en qualsevol cas, mentre que els nodes que tenen associades condicions d'actualització de vistes, sols seran considerats per determinades transaccions. És per aquest motiu, que cal diferenciar el que anomenem Node Actiu del node no actiu. Un Node Actiu és aquell node que té associada una condició que ha de ser comprovada per una transacció T.

Definició 6.17: Donada una traducció TC composada per una transacció T i un conjunt de condicions C, diem que un node del Graf de Precedències és un *Node Actiu* si la condició associada al node és una condició de restricció d'integritat o si es correspon a una condició del conjunt C.

Exemple 6.24: Donat el Graf de Precedències de l'exemple 6.18 i la traducció composada per una transacció T qualsevol i el conjunt de condicions C = {← Treb(p,c) ∧ μTreb(p,c,c1) ∧ c1≠c ∧ Cont(p,c) ∧ ¬μCont(p,c,c1)} el conjunt de nodes actius està format pels nodes C1...C15 i D2. Els primers es corresponen als nodes que tenen associades condicions de restriccions d'integritat i el darrer node té associada la condició d'actualització de vistes del conjunt C.

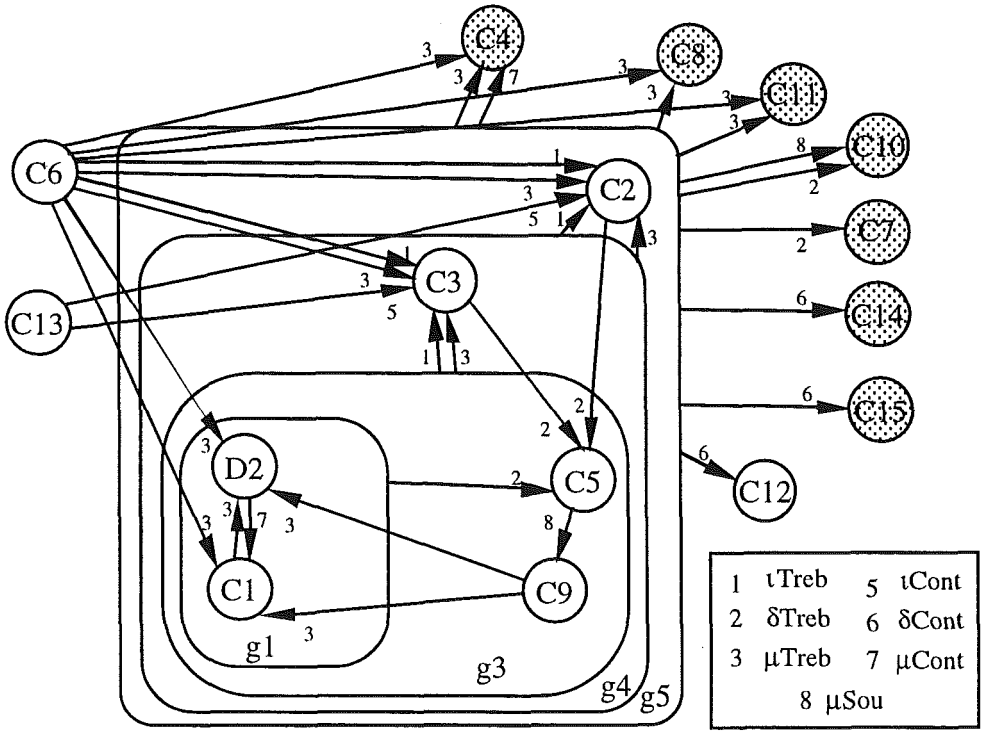
□

Tenint en compte el concepte de node actiu podem definir el Graf de Precedències Actiu, el qual sols inclou els nodes actius i les arestes entre aquests. És a dir, el subconjunt del Graf de Precedències que és rellevant per a cada transacció T.

Definició 6.18: Donat un Graf de Precedències GP i un conjunt de nodes actius, diem que un *Graf de Precedències Actiu* és el subgraf de GP composat únicament per nodes actius i les precedències entre aquests.

Exemple 6.25: Tenint en compte els nodes actius de l'exemple 6.24, el Graf de Precedències Actiu es correspondrà al representat a la figura 6.6. Observi's que en aquest Graf no hi apareixen els nodes associats a condicions d'actualització de vistes que la transacció en curs no ha de comprovar (D1, D3, D4, M1). Per altra banda, el conjunt de condicions de comprovació és Cc=∅.

□



6.6 Graf de Precedències Actiu de l'exemple 6.24

Així doncs, el procediment de manteniment de la consistència, no ha de considerar directament el Graf de Precedències obtingut en temps de definició. Sinó que, per a cada transacció T, s'ha de generar el Graf de Precedència Actiu associat. Aquest graf serà el que realment determinarà l'ordre entre les condicions que la transacció T haurà de comprovar.

En molts casos, durant el procés de comprovació de la consistència d'una transacció poden aparèixer noves condicions d'actualització de vistes que cal comprovar i, per tant, que cal activar en el Graf de Precedències Actiu. Per a incloure-les, únicament cal afegir (activar) en el Graf de Precedències Actiu el node, d Graf de Precedències, que té associada aquesta condició. Caldrà afegir també en el Graf de Precedències Actiu, les precedències d'aquest node amb altres nodes actius. Observi's doncs, que mentre el Graf de Precedències es defineix completament en temps de definició, el Graf de Precedències Actiu és dinàmic i es va definint i completant al llarg del procés de manteniment de la consistència, en temps d'execució.

Condicions associades a cada node

En la definició del Graf de Precedències, a cada node del graf hem associat una única condició: una condició de restricció d'integritat o una condició d'actualització de vistes. En el cas del nodes (C_i) que tenen associada una condició de restricció d'integritat, aquesta es correspon al cos d'una de les regles d'esdeveniment d'inserció d'un predicat d'inconsistència. En canvi en els nodes (A_i, I_i, D_i, M_i) la condició d'actualització de vistes associada es correspon al cos d'una regla que defineix un predicat auxiliar o de transició; o al cos d'una regla d'esdeveniment d'inserció, d'esborrat o de modificació d'un predicat derivat.

En el cas del Graf de Precedències Actiu, el nombre de condicions associades a cada node pot ser superior a una. En el cas de les condicions de restricció d'integritat, cada node etiquetat com a C_i conté una única condició que és la mateixa que en el cas del Graf de Precedències. Però pels nodes actius que tenen associades condicions d'actualització de vistes la situació és diferent.

Aquests nodes (A_i, I_i, D_i, M_i) poden tenir associades diferents condicions específiques, totes elles corresponents a diferents instanciacions i avaluacions de la condició més general que aquest node té associada en el Graf de Precedències. Aquest fet és degut, com ja s'ha comentat en la secció 6.3, a que les condicions d'actualització de vistes es generen en temps d'execució, ja que venen determinades per la petició d'actualització inicial U. En el Graf de Precedències, s'han considerat unes condicions més generals que sí són conegudes en temps de compilació, i que cada una de les possibles condicions d'actualització de vistes generades en temps d'execució es correspondran a una instanciació o avaluació parcial d'aquestes condicions més generals. Així doncs, es pot donar el cas que en temps d'execució es generin dues o més instanciacions/avaluacions de la mateixa condició general i, per tant, caldrà associar aquestes condicions específiques al mateix node actiu.

Exemple 6.26: Consideri's el node D2 del Graf de Precedències i el conjunt de condicions C obtingut per un procés d'actualització de vistes que inclou les condicions següents:

$\leftarrow$ Treb(Pere,Uv) $\wedge$ μTreb(Pere,Uv,c1) $\wedge$ c1$\neq$Uv $\wedge$ Cont(Pere,Uc) $\wedge$ $\neg\mu$Cont(Pere,Uv,c1)

$\leftarrow$ Treb(Ana,Gu) $\wedge$ μTreb(Ana,Gu,c1) $\wedge$ c1$\neq$Gu $\wedge$ Cont(Ana,Gu) $\wedge$ $\neg\mu$Cont(Ana,Gu,c1)

El node D2 del Graf de Precedències Actiu contindrà aquestes dues condicions. Així doncs, al considerar aquest node en el procés de manteniment de la consistència haurem d'assegurar que les dues condicions no estiguin violades, i si és necessari caldrà reparar les condicions que ho requereixin.

□

El tractament que haurem d'aplicar als dos tipus de nodes serà el mateix. La única diferència estarà en que en un cas sols cal comprovar i mantenir una condició, i en l'altre cas, caldrà comprovar i mantenir totes i cada una de les condicions associades a aquest node.

Marcatge dels nodes actius

El primer pas del mecanisme de comprovació de la consistència serà la obtenció del Graf de Precedències Actiu a partir del Graf de Precedències. Un cop obtingut aquest graf, les precedències entre els nodes actius seran les que determinin l'ordre en què cal comprovar les condicions associades als diferents nodes.

Durant el procés de manteniment de la consistència, cal portar un control de quins nodes ja han estat considerats i quines condicions de quins nodes resten pendents de ser comprovades. Per a portar aquest control, a cada un dels nodes (actius) del Graf de Precedències Actiu se li assigna una marca per a indicar que encara no han estat considerats, o se'ls desmarca, per a indicar que les condicions associades ja han estat comprovades (i reparades).

En el cas particular de nodes que tenen associades més d'una condició d'actualització de vistes, tindrem una marca específica per a cada una d'aquestes condicions. Així doncs, un d'aquests nodes estarà desmarcat si i sols si, totes les condicions associades estan també desmarcades. Per a la resta de nodes, sols hi ha una marca associada a la condició del node.

La gestió del marcatge dels nodes consistirà bàsicament en: en primer lloc, un cop obtingut el Graf de Precedències Actiu inicial, es marcaran tots els nodes actius. Durant tot el procés de manteniment de la consistència, al considerar un node marcat N, es comprovaran (i repararan) totes les condicions associades i, un cop finalitzada aquesta comprovació, es desmarcarà el node. Abans de tractar el següent node, tenint en compte les reparacions realitzades i les precedències del graf, caldrà marcar els nodes N' successors a N, tal que existeix una precedència $N \rightarrow N'$ degut a E, si i sols si E es un esdeveniment que s'ha generat al reparar les condicions associades al node N.

Cal tenir en compte que les marques dels nodes són binàries en el sentit que al marcar un node que ja està marcat, no comporta cap canvi en la marca del node. Si el node a marcar N' té més d'una condició associada, caldrà marcar totes les seves condicions. De la mateixa manera, el desmarcar un node es correspon a desmarcar cada una de les condicions associades, i per tant, totes elles tenen que haver estat comprovades (i reparades) satisfactòriament.

En el moment en que tots els nodes (actius) del Graf de Precedències Actiu estiguin desmarcats, voldrà dir que totes les condicions del graf han estat comprovades i reparades satisfactòriament i, per tant, la transacció obtinguda es correspon una solució de la petició d'actualització inicial U.

Una possible millora en la gestió de marques del Graf de Precedències Actiu, consisteix en fer un marcatge selectiu dels nodes tenint en compte un mètode que identifiqui quan una consulta (o condició) és independent a una actualització [LS93, LL96]. En aquest cas, sols es marcaria un node si la condició associada pot ser violada per la transacció considerada. De la mateixa manera, en el marcatge inicial del graf, sols es marcarien els nodes actius que són

poden ser violats per la transacció T. Amb la utilització d'algun d'aquests mètodes es reduiria considerablement el nombre de condicions a comprovar i ens podríem beneficiar de l'eficiència pròpia d'aquests mètodes al comprovar quines condicions són realment violades i quines no. De totes formes, tenint en compte la informació referent als violadors potencials d'una condició podríem fer un marcatge més intel·ligent dels nodes.

Criteris de selecció del següent node a tractar

El Graf de Precedències Actiu defineix implícitament un ordre entre els diferents nodes que el composen i, per tant, l'ordre en que s'han de comprovar les condicions que ens asseguren la consistència d'una transacció T, de forma que cada node del graf es consideri el menor nombre de vegades. Per això, caldrà definir criteris per a seleccionar, després de tractar un node, el següent node a considerar.

El mecanisme per a seleccionar els nodes a tractar amb un ordre concret, està determinat per la pròpia estructura del Graf de Precedències Actiu i pels criteris descrits a continuació:

1. Un node és candidat a ser seleccionat com a següent node a visitar, sempre que estigui marcat, és a dir, sempre que encara tingui condicions pendents de ser comprovades.
2. Un node candidat a ser seleccionat com a següent node a ser visitat, ha de tenir tots els nodes precedents desmarcats. D'aquesta manera assegurem que una condició no és comprovada fins que totes les condicions, les reparacions de les quals la poden violar, ja han estat comprovades. Aquesta afirmació serà certa sempre que aquesta condició no formi part de cap cicle dins el Graf de Precedències Actiu, i sempre que no s'activi cap nou node que sigui predecessor a algun dels nodes ja tractats.
3. Si el darrer node tractat forma part d'un subgraf (cicle), no es podran seleccionar nodes externs al subgraf fins que tots els nodes del subgraf estiguin desmarcats. A més a més, solament s'han de considerar les precedències entre nodes del subgraf.
4. De tots els nodes candidats a ser seleccionats com a següent node a visitar, es preferible seleccionar amb màxima prioritat aquells que tenen associades condicions de comprovació. El fet de seleccionar nodes amb condicions de comprovació millora l'eficiència del procés de manteniment de la consistència. Si una condició de comprovació es viola, s'ha de rebutjar la transacció T i iniciar de nou el procés amb una nova transacció. Com més aviat es comprovin aquestes condicions de comprovació, menys feina innecessària haurem de fer tractant altres condicions, ja que finalment aquesta condició pot rebutjar la transacció actual.
5. Si un cop aplicats els criteris anteriors, encara hi ha més d'un candidat a ser el següent node a ser visitat, és preferible seleccionar aquells nodes que no tenen cap node precedent, ja que en aquests casos, no hi ha cap condició en el graf, la reparació de la qual pugui violar la condició d'aquest node.

6. En cas de tenir encara més d'un candidat, cal escollir-ne un qualsevol entre els nodes candidats.

Quan el node seleccionat és un node subgraf, el tractament d'aquest node es correspon a una crida recursiva del mecanisme de manteniment de la consistència als nodes que composen el cicle associat al subgraf. En aquest cas, cal seleccionar un node qualsevol del cicle que estigui marcat. Els successius nodes que s'aniran seleccionant, han de formar part del cicle, i no es podrà seleccionar cap node extern al subgraf fins que s'hagin mantingut totes les condicions de tots els nodes del cicle, és a dir, fins que tots els nodes del subgraf no estiguin desmarcats.

Cal tenir en compte que, degut a la reparació d'alguna condició que requereix un procés d'actualització de vistes, pot ser necessari activar nous nodes en el Graf de Precedències Actiu. En aquests casos, pot donar-se la particularitat de que calgui afegir nous nodes actius al Graf o marcar de nou nodes que ja s'havien desmarcat i que, precedeixen a nodes que ja han estat tractats. En aquests casos, al igual que en els casos dels cicles, les condicions associades a un node poden ser comprovades més d'una vegada.

Exemple 6.27: Consideri's el Graf de Precedències Actiu de l'exemple 6.24 on tots els nodes estan marcats. Per a mostrar l'ordre en que es seleccionaran els diferents nodes, considerarem que no es viola cap condició del graf i per tant, els nodes seleccionats es desmarquen i no es propaguen les marques cap als successors del node visitat. Tenint en compte els criteris anteriors, un possible ordre de selecció dels nodes del Graf de Precedències Actiu podria ser el següent: C13, C6, g5, C4, C7, C8, C10, C11, C14, C15, C12.

Els primers nodes candidats a ser seleccionats han de ser els nodes C13 i C6, ja que són els únics que no tenen cap node predecessor marcat. Qualsevol dels dos nodes pot ser seleccionat en primer lloc. Per exemple, seleccionem el node C13. El següent node ha de ser el node C6, ja que és predecessor dels nodes del subgraf g5. Un cop desmarcat C6 cal seleccionar el node g5 ja que la resta de nodes el tenen com a predecessor. El tractament del subgraf g5 es comenta en el paràgraf següent. Un cop tractat el subgraf g5, els nodes candidats a ser seleccionats són els nodes C4, C7, C8, C10, C11, C14, C15, C12. Observi's que els nodes associats a condicions de comprovació (C4, C7, C8, C10, C11, C14, C15) són els que es seleccionaran abans del node C12, ja que aquest node té associada una condició de generació. L'ordre de selecció entre aquest conjunt de nodes és aleatori ja que no tenen definides precedències entre ells. Un cop seleccionats tots ells, ja podem seleccionar el darrer node C12.

Al seleccionar el node subgraf g5, ens trobem que està compost pels nodes g4 i C2. Els dos són predecessors entre sí, per tant, no tenim un criteri específic a aplicar i podem seleccionar un dels dos, per exemple g4. Al seleccionar g4, ens trobem amb la mateixa situació amb els nodes g3 i C3. I el mateix passa amb els subgrafs g3 i g1. Al seleccionar el subgraf g1, podem seleccionar els nodes C1 o D2 indistintament. Un cop desmarcats els dos, no resten nodes marcats dins el subgraf g1 i ja podem seleccionar nodes externs al subgraf. Per tant, procedirem

a la selecció del node C5 i després el seu successor C9. Desmarcats tots ells seleccionem el node C3 i finalment el C2. En aquest punt tots els nodes del subgraf g5 estan desmarcats i ja podem seleccionar nodes externs al subgraf.

□

Els diferents ordres que es puguin definir a partir de l'estructura pel Graf de Precedències Actiu són equivalents a nivell d'eficiència, ja que tots els nodes marcats del graf hauran de comprovar-se en un moment o altre. Només seria possible identificar-ne un com a més eficient si, en el moment de seleccionar un node, tinguéssim informació referent a l'existència d'alguna condició en un node que no serà possible reparar i per tant, aquest node seria el candidat a comprovar en primer lloc. Al no disposar d'aquesta informació, qualsevol d'aquests ordres és vàlid per al manteniment de la consistència i no es pot preveure quin d'ells serà més eficient.

Conjunt auxiliar de condicions de comprovació Cc

A la secció 6.3.3 s'ha introduït una optimització del Graf de Precedències consistent en reduir el nombre de nodes que hi ha representats en el Graf de Precedències, eliminant tots els nodes etiquetats com a B_i (i les precedències en que participen). Aleshores, totes les condicions que es vagin generant en temps d'execució que haurien d'estar associades a aquests nodes cal acumular-les en el Conjunt de Condicions de Comprovació anomenat Cc.

Aquests nodes contenen una o vàries condicions d'actualització de vistes que tenen el format següent: $\leftarrow$ Ev, on Ev es correspon a un esdeveniment bàsic. Aquestes condicions són condicions de comprovació i únicament poden ser violades quan l'esdeveniment Ev pertanyi a la transacció T.

El tractament que proposem per aquest conjunt de condicions és el següent: cada cop que s'afegeixin nous esdeveniments a la transacció T; cada cop que hi hagi noves incorporacions al conjunt Cc; i un cop a l'inici del procés de manteniment de la consistència, es comprovaran cada una de les condicions del conjunt Cc. En cas de violar-ne alguna, es rebutja la transacció T sense necessitat de considerar altres nodes del Graf de Precedències Actiu. En cas de no violar-ne cap, es procedirà a seleccionar una node del Graf de Precedències Actiu aplicant el criteris anteriors.

La gestió d'aquestes condicions mitjançant el conjunt Cc, és especialment avantatjosa respecte a un tractament d'aquestes condicions com a nodes B_i en el Graf de Precedències Actiu. El tractament d'aquestes condicions en el conjunt Cc permet descartar transaccions que violen aquestes condicions el més aviat possible, i a la vegada, permet evitar tenir de comprovar i reparar les condicions de tots els nodes marcats que siguin predecessors als nodes B_i en el Graf de Precedències Actiu. El possible inconvenient d'aquest tractament específic del conjunt Cc rau en el fet que cada una d'aquestes condicions pot ser comprovada diverses vegades. Però, per altra banda, cal tenir en compte que aquesta comprovació té un cost molt reduït ja que sols cal comprovar si l'esdeveniment bàsic Ev pertany o no a la transacció T.

Cicles entre nodes

Com ja ha estat comentat en la secció 6.3.4, en la definició del Graf de Precedències, es pot donar el cas que existeixin cicles entre un subconjunt de nodes d'aquest graf. Aquests nodes s'agrupen en un tipus de node especial que anomenem subgraf. De forma similar, en la generació del Graf de Precedències Actiu, alguns d'aquests cicles poden seguir existint.

Els nodes subgraf, dins del Graf de Precedències Actius, requereixen un tractament diferent al nodes que contenen condicions. Els nodes subgrafs es tractaran com a grafs de precedències actius, i el mateix procediment pel manteniment de la consistència aplicat al Graf de Precedència Actiu s'aplicarà de forma recursiva a cada un dels subgraf que existeixin.

És importat observar, que encara que, en temps de definició, les precedències entre el nodes d'un subgraf defineixen un cicle, aquests cicles no sempre es corresponen a cicles infinits en temps d'execució.

En primer lloc podem comprovar que donada una condició qualsevol C, el nombre de vegades que es pot violar (amb una substitució α), reparar i tornar a violar (amb aquesta mateixa substitució α) és sempre finit. Per altra banda, també es pot comprovar que una condició C pot ésser violada per un nombre finit de substitucions.

Suposem una condició C en la que els esdeveniments que apareixen al seu cos són únicament esdeveniments bàsics. Aquesta condició té un nombre limitat de violadors potencials i reparadors potencials. Els violadors potencials es corresponen als esdeveniments bàsics positius i el reparadors potencials són els esdeveniments bàsics que apareixen de forma negada. Suposem que una transacció T viola aquesta condició amb una substitució α. Per a fer-ho, la transacció T ha d'incloure una substitució α de tots els violadors potencials de C. Per a reparar-la, cal afegir la substitució α d'un reparador potencial qualsevol de C a la transacció T. Observi's que un cop reparada la violació α de la condició C, mai més podrà esdevenir violada de nou amb la mateixa substitució α. En tot cas, aquesta condició es violarà amb una substitució diferent ($\beta \neq \alpha$).

De forma similar, una condició C definida per esdeveniments derivats també tindrà un conjunt finit de violadors i reparadors potencials, els quals es poden identificar a partir del Graf de Dependències entre esdeveniments. En aquest cas, per a violar la condició C, la transacció T ha d'incloure una substitució α d'un subconjunt del seus violadors potencials, de forma que indueixin tots els esdeveniments que apareixen de forma positiva al cos de la condició. Per a reparar-la, cal afegir a la transacció T la substitució α d'un subconjunt de reparadors potencials de C que permetin induir un esdeveniment que apareix negat e.. el cos de la regla o que impedeixin induir un esdeveniment derivat que apareix de forma positiva. Observi's que un cop reparada la violació α de la condició C, en aquest cas, sí es possible tornar a violar la condició C amb la mateixa substitució. Tan sols és necessari afegir a la transacció T nous violadors potencials que impedeixin induir l'esdeveniment derivat negat que havia reparat la condició C.

Però de totes maneres, degut a que el nombre de reparadors i violadors potencials és limitat, el nombre de vegades que es pot violar la condició C (amb la substitució α) i reparar-la de nou és limitat. Per altra banda, aquesta condició sempre podrà esdevenir violada per una substitució diferent ($\beta \neq \alpha$).

El nombre de violacions diferents que pot induir una transacció T és sempre finit, ja que el nombre d'esdeveniments que conté també ho és. A més a més, al considerar dominis finits, l'extensió dels predicats existencials i els esdeveniments d'aquests és també finita.

Com veurem en l'exemple següent, els cicles definits en el Graf de Precedències Actiu de l'exemple 6.25 no donaran lloc a cicles infinits en temps d'execució. En aquest cas, els cicles no són infinits per dues raons diferents a les justificacions anterior: per una banda perquè un reparador potencial d'una condició també ho és d'una altre condició i, per l'altra, perquè el cicle requereix l'execució d'esdeveniments mútuament exclusius.

Exemple 6.28: Considerem els cicles que apareixen en el Graf de Precedències Actiu de l'exemple 6.25:

Cicle g1:

$C1 \rightarrow D2$ degut a μTreb

$D2 \rightarrow C1$ degut a μCont

(C1) $\leftarrow Emp(\underline{p},c) \wedge \neg\delta Emp(\underline{p},c) \wedge \neg Aux3(\underline{p},c) \wedge Edat(\underline{p}) \wedge \delta Edat(\underline{p})$

(D2) $\leftarrow Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c1) \wedge c1 \neq c \wedge Cont(\underline{p},c) \wedge \neg\mu Cont(\underline{p},c,c1)$

(A3) $Aux3(p,c) \leftarrow \mu Emp(p,c,c1)$

(M1) $\mu Emp(\underline{p},c,c1) \leftarrow Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c1) \wedge Cont(\underline{p},c) \wedge \mu Cont(\underline{p},c,c1) \wedge c \neq c1$

Observi's que en aquest cas l'única forma de reparar D2 és mitjançant l'esdeveniment μCont(p,c,c1). Aquest esdeveniment és a la vegada un violador i reparador de C1, i per tant, al reparar D2 s'indueix l'esdeveniment μEmp(p,c,c1) i la condició C1 esdevé automàticament reparada. Com que aquest esdeveniment derivat no es pot fer fals, aleshores, no es pot produir cap cicle entre les dues condicions.

Cicle g3:

$C1 \rightarrow C5$ degut a δTreb

$C5 \rightarrow C9$ degut a μSou

$C9 \rightarrow C1$ degut a μTreb

Observi's que aquest cicle mai es podrà donar en temps d'execució, ja que els esdeveniments δTreb(p,c) i μTreb(p,c,c1) són mútuament exclusius, i per tant, no podran pertànyer al mateix temps a la mateixa transacció T.

Cicle g4:

$C2 \rightarrow C5$ degut a δTreb

$C5 \rightarrow C9$ degut a μSou

$C9 \rightarrow C2$ degut a ιTreb

$C9 \rightarrow C2$ degut a μTreb

Cicle g5:

$C3 \rightarrow C5$ degut a δTreb

$C5 \rightarrow C9$ degut a μSou

$C9 \rightarrow C3$ degut a ιTreb

$C9 \rightarrow C3$ degut a μTreb

De la mateixa manera que en el cas anterior, aquests cicles no es donaran mai en temps d'execució, ja que els esdeveniments ιTreb(p,c) i μTreb(p,c,c1) són mútuament exclusius respecte a l'esdeveniment δTreb(p,c).

□

6.4.2 Procediment pel manteniment de la consistència

Un cop establertes aquestes consideracions bàsiques, ja estem en situació de descriure en detall el mecanisme proposat pel manteniment de la consistència.

El punt de partida per a mantenir la consistència d'una transacció és: un Graf de Precedències GP i una traducció, composada per una transacció T i un conjunt de condició C. El resultat d'aquest procés serà el conjunt de totes les solucions mínimes S_i ($T \subseteq S_i$) que no violen cap condició del Graf de Precedències GP.

En primer lloc i tenint en compte les condicions del conjunt C i el Graf de Precedències PG, es genera el Graf de Precedències Actiu associat a la transacció T. Aquesta generació consisteix en seleccionar i activar tots el nodes i arestes que tenen associades condicions de restricció d'integritat i els nodes (amb les arestes adjacents) que tenen associades alguna de les condicions del conjunt C. Un cop generat el Graf de Precedències Actiu es marquen tots els seus nodes. Al mateix temps, es genera el conjunt de Condicions de Comprovació Cc i es comprova que la transacció T no violi cap d'aquestes condicions de comprovació. En cas de fer-ho, finalitza el procés de manteniment de la consistència generant la transacció final $S=\varnothing$. En cas contrari, es procedeix a mantenir les condicions dels nodes del Graf de Precedències Actiu i les del conjunt Cc.

Mentre en el Graf de Precedències Actiu restin nodes marcats, cal seleccionar un d'aquests nodes seguint els criteris definits en la secció anterior. Sigui C el node seleccionat i suposem que no es correspon a cap node subgraf. Per a tractar aquest node, cal obtenir totes les violacions de totes les condicions que té associades el node. Per a cada una d'aquestes violacions cal obtenir-ne totes les possibles formes de reparar-les. Si alguna d'aquestes reparacions requereix l'actualització d'algun fet derivat, caldrà crid al mòdul d'Actualització de Vistes per obtenir-ne totes les possibles traduccions. Si solament es requereixen actualitzacions de fets bàsics, aquests s'obtenen directament de la pròpia definició de la condició violada. En qualsevol cas, sigui R una de les transaccions que reparen totes les condicions associades al node C, i sigui Rc el conjunt de condicions addicional associat a la transacció

reparadora R. Un cop reparat satisfactòriament el node C, cal desmarcar-lo i marcar els nodes successors de C que tenen una aresta etiquetada amb algun dels esdeveniments de R. En cas de no poder-se reparar satisfactòriament totes les condicions del node C, cal rebutjar la transacció T per inconsistent.

La transacció resultant del tractament del node C es correspondrà a T' = T ∪ R. El nou Graf de Precedències Actiu és igual a l'anterior Graf amb el node C desmarcat, les noves marques en els nodes successors de C, i amb nous nodes actius (i marcats). Aquests nous nodes són els que tenen associades les condicions del conjunt Rc. Les condicions del conjunt de condició Rc de tipus B_i s'inclouran al conjunt Cc, i es comprovarà que els esdeveniments de R no violen cap condició d'aquest conjunt Cc.

En aquest moment, si encara resten nodes marcats en el Graf de Precedències Actiu, es seleccionarà un nou node i es repetirà el procés anterior. En el cas de tenir tots els nodes del Graf de Precedències Actiu desmarcats, la transacció S=T' es correspon a una solució de la petició inicial d'actualització.

En el cas en què el node seleccionat C es correspongui a un node subgraf, el tractament d'aquest node consistirà en l'execució recursiva d'aquest mateix mecanisme, però sols en l'àmbit del subgraf de precedències cíclic associat al node. Un cop executada aquesta crida, cal desmarcar el node C i caldrà també marcar i activar els nodes externs al subgraf C, tenint en compte les possibles reparacions realitzades per a mantenir els nodes del cicle.

6.4.3 Algorisme pel manteniment de la consistència

Per a realitzar aquest procés de manteniment de la consistència, hem definit un procediment Manteniment_Consistència(T, C, GP) que donada una transacció T, comprova que aquesta sigui consistent respecte a les condicions d'un Graf de Precedències GP i un conjunt de condicions C. El resultat és el conjunt de totes les solucions que satisfan T i asseguren aquesta consistència. A continuació es presenta l'algorisme associat a aquest procediment:

```
Procediment Manteniment_Consistència(T, C, GP)
  /* entrada: una transacció T, un conjunt de condicions C i un Graf de Precedències GP */
  /* sortida: imprimeix totes les transaccions que satisfan T i les condicions del Graf de Precedències*/
GPA := Generar_Graf_Actiu(GP, T, C);
Cc := Generació_Conjunt_Condicions_Comprovació(C);
Rebutjar := Comprovació_Cc(T, Cc);
si no Rebutjar llavors
        SS := Manteniment_Consistència_Graf(T, GPA, Cc, ∅);
        per cada (T, C) ∈ SS fer
                imprimir(T)
        fi per cada
fi si
Return;
```

Aquest procediment és l'encarregat de generar el Graf de Precedències Actiu GPA a partir del Graf de Precedències GP i de comprovar la consistència de la transacció T respecte aquest graf. La funció Generar_Graf_Actiu genera i marca el Graf de Precedències Actiu associat a la transacció T tenint en compte el conjunt de condició C i el Graf de Precedències GP. Abans d'iniciar la comprovació de les condicions associades a cada un dels nodes del Graf de Precedències Actiu, es genera el conjunt de condicions de comprovació Cc i es comprova que no n'hi hagi cap de violada. En cas de violar-se'n alguna, es rebutja la transacció T i finalitza el procediment. En el cas que la transacció T no en violi cap, es procedeix a comprovar la consistència de la transacció T respecte al Graf de Precedències Actiu amb la funció Manteniment_Consistència_Graf. El resultat d'aquesta funció és el conjunt SS de totes les traduccions (transacció, condicions) que satisfan la petició d'actualització inicial U i les condicions del Graf de Precedències Actiu. Si aquest conjunt és buit, vol dir que la transacció T és inconsistent, ja que no pot satisfer a la vegada la petició inicial d'actualització U i les restriccions d'integritat de la base de dades. En cas contrari, es mostren a l'usuari les diferents solucions obtingudes en el conjunt SS.

La funció del procediment anterior que assegura la consistència de la transacció T tenint en compte el Graf de Precedències Actiu, és la funció Manteniment_Consistència_Graf(T, GPA, Cc, C). Els paràmetres d'aquesta funció es corresponen a la transacció T; el Graf de Precedències Actiu GPA i el Conjunt de Condicions de Comprovació Cc. El graf GPA i el conjunt Cc contenen les condicions que asseguren la consistència de la transacció T. El conjunt de condicions addicional C és un paràmetre necessari per a la gestió de la recursivitat de la pròpia funció, en el cas que GPA tingui cicles. El resultat de la funció conté el conjunt de totes les possibles transaccions mínimes S_i tal que $T \subseteq S_i$ i que no violen cap condició del Graf de Precedències Actiu.

L'algorisme associat a aquesta funció és el següent:

```
Funció Manteniment_Consistència_Graf(T, GPA, Cc, C): SS
  /* entrada: una transacció T, el Graf de Precedències Actiu GPA i els conjunts de condicions Cc i C */
  /* sortida: conjunt SS de tots els parells (transacció, condicions) SS={(Ts, Cs)} que satisfan les
     condicions del Graf de Precedències Actiu GPA */

SS := (∅, ∅);
Pendents := [(T, C, GPA, Cc)];
mentre Pendents no buit fer
    Obtenir_transacció(Pendents, T, C, GPA, Cc);
    mentre GPA tingui nodes marcats fer
        Node := Seleccionar_Node_Següent(GPA);
        si Node no és un subgraf llavors
            Desmarcar(Node, GPA);
            violada := Comprovar_Node(Node,
            si violada llavors
                si Comprovació(Node) llavors
                    Obtenir_transacció(Pendents, T, C, GPA, Cc);
```

```
                    sinó  /* Condició de Generació */
                          Prep := Obtenir_totes_peticions_reparació(Node,T);
                          per cada Pr_i ∈ Prep fer
                                si Pr_i conté actualitzacions a fets derivats llavors
                                      Actualització_de_Vistes(Pr_i, TC);
                                sinó  TC := {(Pr_i, ∅)}
                                fi si;
                                per cada (R_i,C_i) ∈ TC fer
                                  GPA' := Activar_i_marcar_nodes(GPA, TC);
                                  Cc' := Afegir_noves_condicions(Cc, C_i);
                                  T' := T ∪ R_i; C' := C ∪ C_i;
                                  Rebutjar := Comprovació_Cc(T', Cc');
                                  si no Rebutjar llavors
                                        Afegir(Pendents, [(T', C', GPA', Cc')]);
                                  fi si;
                                fi per cada;
                          fi per cada;
                          Obtenir_transacció(Pendents, T, C, GPA, Cc);
                    fi si;
              fi si;
        sinó /* Node subgraf */
              Desmarcar(Node, GPA);
              SS_g := Manteniment_Consistència_Graf(T, Node, Cc, C);
              per cada S_g ∈ SS_g fer                    /* S_g = (T_g, C_g) */
                    R_g := T_g - T;
                    S_gr := (R_g, C_g);
                    GPA' := Activar_i_marcar_nodes(GPA, S_gr);
                    Cc' := Afegir_noves_condicions(Cc, C_g);
                    T' := T_g; C' := C ∪ C_g;
                    Rebutjar := Comprovació_Cc(T', Cc');
                    si no Rebutjar llavors
                          Afegir(Pendents, [(T', C', GPA', Cc')]);
                    fi si;
              fi per cada;
              Obtenir_transacció(Pendents, T, C, GPA, Cc);
        fi si;
    fi mentre;
SS := SS ∪ {(T, C)}
fi mentre;
Return(SS);
```

- *Obtenir_transacció* i *Afegir*: permeten obtenir una de les transaccions pendents de finalitzar, i emmagatzemar-ne una que quedarà pendent de finalitzar.
- *Seleccionar_Node_Següent*: és la funció que selecciona el següent node a tractar. Per a fer aquesta selecció, aplica els criteris definits a la secció 6.4.1.
- *Desmarcar*: elimina la marca del node que tractarem a continuació, tant si és un node simple com si és un subgraf.

- *Comprovar_Node*: és la funció encarregada de comprovar si alguna de les condicions associades al Node actual és violada per la transacció T. Aquesta funció es pot implementar utilitzant un mètode de comprovació de restriccions d'integritat.
- *Comprovació*: determina si el node conté condicions de comprovació o de generació.
- *Obtenir_totes_peticions_reparació*: Aquesta funció, donades les condicions associades a un node que són violades per T, genera el conjunt de peticions de reparació per a reparar aquestes violacions. Les peticions de reparació consisteixen en peticions d'actualització de fets bàsics i/o de fets derivats.
- *Actualització_de_Vistes*: Aquest procediment es correspon al mòdul d'Actualització de Vistes definit a la secció 6.5. Donada una petició d'actualització d'un predicat derivat, aquesta funció genera totes les possibles traduccions en actualitzacions de fets bàsics. Cada traducció està composada per una transacció T i un conjunt de condicions C.
- *Activar_i_marcar_nodes*: Donada una traducció TC composada per una transacció T i un conjunt de condicions C, aquest procediment s'encarrega d'afegir en el Graf de Precedències Actiu aquells nous nodes que esdevenen actius degut a les condicions del conjunt C. Aquests nodes es marquen adequadament. En el cas de que el node associat ja sigui actiu, s'afegeix la condició al node i es marca la condició (i el node) oportunament. Aquesta funció també s'encarrega de marcar els nodes successors al node actual.
- *Afegir_noves_condicions*: Aquest procediment afegeix noves condicions al Conjunt de condicions de Comprovació Cc.
- $T' := T \cup R$: La transacció actual T s'estén amb els esdeveniments procedents de la reparació R d'alguna condició. Al mateix temps que es genera la nova transacció T' es comprova que els seus esdeveniments no siguin mútuament exclusius.
- *Comprovació_Cc*: amb aquesta funció es comprova que cap de les condicions de comprovació del conjunt Cc' siguin violades per la transacció T'. En cas de no ser així, caldria rebutjar la transacció actual sense necessitat de fer cap altra comprovació.

L'algorisme de la funció Manteniment_Consistència_Graf té una estructura iterativa, on a cada iteració es realitza la comprovació de la consistència per a una transacció T respecte al graf GPA i el conjunt de condicions de comprovació Cc. Aquestes transaccions, són les que s'han anat generant a partir de la transacció T inicial i a la que s'hi han anat afegint les diferents formes de reparar les condicions associades a un node. Per a cada alternativa de reparació es genera una transacció T' diferent. Aquestes alternatives es van emmagatzemant en una estructura 'Pendents', i s'aniran considerant una darrera l'altra. El tractament de cada una d'aquestes transaccions consisteix a la vegada en un iterador, on a cada iteració es considera un dels nodes del Graf de Precedències Actiu associat. El procés finalitza quan tots els nodes han estat tractats i desmarcats oportunament.

El tractament d'una transacció T es realitza seguint els següents passos. En primer lloc es selecciona el node a tractar amb la funció Seleccionar_Node_Següent. El tractament del node seleccionat és diferent depenent de si aquest node es correspon a un subgraf o no.

En el cas de ser un node simple (no és subgraf), es desmarca aquest node i es comprova si alguna de les condicions associades està violada per T. En cas de ser un subgraf, es procedeix a seleccionar un nou node mentre en restin de marcats. En cas de tenir alguna condició del node violada, es comprova quin tipus de condició és: si és de comprovació, cal descartar aquesta transacció i buscar-ne una de nova a la estructura Pendents; en canvi si és de generació cal reparar-la.

Per a reparar una condició de generació cal obtenir totes les formes de reparar la condició. Aquestes s'emmagatzemen a la variable Prep. Si la petició de reparació Pr conté actualitzacions de fets derivats, es crida al procediment Actualització de Vistes (descrit a la secció 6.5) per tal de traduir aquesta requesta en actualitzacions de fets bàsics. Aquest procediment genera totes les possibles traduccions TC, cada una d'elles composada per una transacció R_i i un conjunt de condició C_i. Si en canvi, la petició de reparació consisteix solament en actualitzacions de fets bàsics, aquests composaran la transacció R_i i el conjunt de condició serà $C_i = \varnothing$.

Un cop obtinguda una reparació, cal activar els nous nodes associats a les condicions del conjunt C_i i marcar els nodes successors a l'actual, obtenint un nou Graf de Precedències Actiu GPA'. Cal també actualitzar el conjunt de condicions de comprovació Cc', estendre la transacció T amb la reparació i acumular les condicions al conjunt C'. La darrera comprovació, abans d'afegir la transacció obtinguda T' a la estructura Pendents, consisteix en comprovar que no es violi cap condició del conjunt Cc'.

Per a continuar amb el procés de comprovació de la consistència de la transacció T respecte al Graf de Precedències Actiu, escollim una de les transaccions T de la estructura Pendents i comencem de nou amb la selecció d'un nou node a tractar.

En el cas de haver seleccionat un node subgraf, es desmarca aquest node i es realitza una crida (recursiva) de la funció Manteniment_Consistència_Graf, però en aquesta cas, limitada al subgraf de precedències actiu associat. El resultat d'aquesta crida seran el conjunt de parelles (transacció, condicions) que satisfan les condicions dels nodes del subgraf. Observi's que el tractament posterior que es fa de les traduccions obtingudes és equivalent a les reparacions que s'obtindrien d'un node simple.

En el conjunt auxiliar C es van acumulant totes les condicions associades a la transacció T i que serveixen per activar nous nodes externs a un subgraf. En la crida recursiva d'aquesta funció, solament es consideren els nodes actius que formen part del cicle. Tots els nodes externs al subgraf que calgui activar, s'activaran un cop s'ha finalitzat l'execució de la crida recursiva. És per aquesta raó, per la qual aquestes condicions es retornen pel paràmetre C de la crida, junt amb la transacció T.

Un cop desmarcats tots els nodes del graf de Precedències Actiu, la transacció T i el conjunt C s'acumulen en el conjunt SS. Aquest conjunt es correspon al conjunt de totes les solucions que satisfan la petició inicial d'actualització U i no violen cap restricció d'integritat. Aquest conjunt SS es retorna com a resultat, un cop tractades totes les transaccions acumulades a la estructura Pendents.

Exemple 6.29: Consideri's la base de dades de l'exemple 3.1, la seva base de dades augmentada A(D), el Graf de Precedències definit a l'exemple 6.18 i la base de dades extensional següent:

Treb(Mercè, Gu) Cont(Mercè, Gu) Edat(Mercè) Numss(Mercè,12345)

Sigui T={δEdat(Mercè)} la transacció a aplicar a la base de dades i el conjunt de condicions C=∅. El Graf de Precedències Actiu GPA associat a aquesta transacció és el representat a la figura 6.7 i el conjunt de condicions de comprovació Cc=∅.

L'execució de la funció Manteniment_Consistència_Graf(T, GPA, Cc, ∅) es descriu de forma resumida mitjançant una taula 6.1. Cada columna es correspon a un node actiu del Graf de Precedències Actiu. La columna de la dreta indica quin és el contingut de la transacció T'. Les marques de cada node s'indiquen mitjançant el símbol 'x', i amb '**X**' el node seleccionat. En cada fila de la taula es mostren quins canvis hi ha hagut respecte a les marques dels diferents nodes del Graf de Precedències Actiu i, amb *cursiva* s'indiquen quins nous esdeveniments han estat incorporats a la transacció T' com a resultat de la reparació de la condició del node seleccionat.

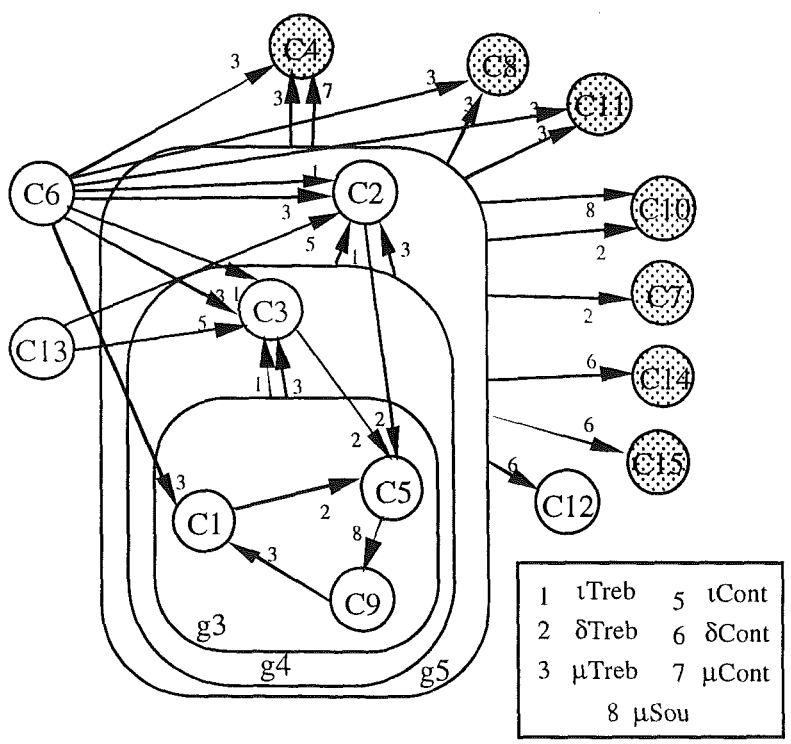


Fig.6.7 Graf de Precedències Actiu de l'exemple 6.29

A la taula 6.1, solament es resumeix el procés per a obtenir una solució a la traducció T. El mecanisme per a obtenir la resta de solucions és similar al descrit a la taula.

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	g3	g4	g5	Transacció
(1)	x	x	x	x	x	**X**	x	x	x	x	x	x	x	x	x	x	x	x	δEdat
(2)	x	x	x	x	x		x	x	x	x	x	x	**X**	x	x	x	x	x	δEdat
(3)	x	x	x	x	**X**		x	x	x	x	x	x		x	x	**X**	**X**	**X**	δEdat
(4)	x	x	x	x			x	x	**X**	x	x	x		x	x	**X**	**X**	**X**	δEdat
(5)	**X**	x	x	x			x	x		x	x	x		x	x	**X**	**X**	**X**	δEdat, *δTreb*
(6)		x	x	x	**X**		x	x		x	x	x		x	x	**X**	**X**	**X**	δEdat, δTreb
(7)		x	**X**	x			x	x		x	x	x		x	x		**X**	**X**	δEdat, δTreb
(8)		**X**		x			x	x		x	x	x		x	x			**X**	δEdat, δTreb
(9)				**X**			x	x		x	x	x		x	x				δEdat, δTreb
(10)							**X**	x		x	x	x		x	x				δEdat, δTreb
(11)								**X**		x	x	x		x	x				δEdat, δTreb
(12)										**X**	x	x		x	x				δEdat, δTreb
(13)											**X**	x		x	x				δEdat, δTreb
(14)												x		**X**	x				δEdat, δTreb
(15)												x			**X**				δEdat, δTreb
(16)												**X**							δEdat, δTreb, *δNumss*

Taula 6.1. Resultat execució Manteniment_Consistència_Graf exemple 2.29

En els passos (1) i (2), s'han seleccionat els nodes C6 i C13 ja que no tenen predecessors. Les condicions associades no esdevenen violades per T i, per tant, es desmarquen els nodes i es procedeix a seleccionar-ne un altre. En el pas (3) es selecciona el node subgraf g5 i s'executa una crida recursiva de l'algorisme. Dins els subgraf es selecciona el node subgraf g4 i es procedeix de forma similar. Dins aquest subgraf és el node g3 i, finalment, dins aquest subgraf es selecciona el node C5. La condició C5 es desmarca ja que no està violada per T. El següent node a tractar, dins el subgraf g3 és el node C9 que tampoc es viola i es desmarca. Al pas (5) es selecciona el node C1. En aquest cas, la transacció T viola la condició associada i cal repara-la. Per a fer-ho, cal executar el procediment Actualització_de_Vistes amb la petició d'esborrat δEmp(Mercè, Gu). El resultat d'aquesta petició de reparació genera quatre traduccions alternatives:

T_1={δTreb(Mercè, Gu)} C_1=∅;

T_2={μTreb(Mercè, Gu, P3)} C_2={←μCont(Mercè, Gu, P3)};

T_3={δCont(Mercè, Gu)} C_3=∅;

T_4={μCont(Mercè, Gu, P3)} C_4={←μTreb(Mercè, Gu, P3)}

En aquest exemple sols considerarem la traducció $TC_1 = (T_1, C_1)$.

Un cop reparada la condició C1, cal desmarcar-la i marcar de nou el node C5, ja que l'esdeveniment δTreb és un potencial violador de C5. Altres condicions (nodes) que poden esdevenir violats per aquesta reparació són els nodes C7 i C10. Aquests nodes es marcaran al

finalitzar el tractament del subgraf g5. En el pas (6) es comprova que aquesta condició no es viola i es desmarca. En aquest punt, tots els nodes que composen el subgraf g3 estan desmarcats, per tant, finalitza la crida recursiva associada a aquest subgraf i ja podem seleccionar nodes del subgraf g4. Als passos (7) i (8), els nodes seleccionats no es violen i finalitzem el tractament dels subgraf g4 i g5 respectivament. En aquest punt, caldria marcar els nodes C7 i C10 però ja estan marcats. Des del pas (9) fins al pas (15) es seleccionen nodes que contenen condicions de comprovació i que han de seleccionar-se abans de qualsevol altre node que contingui condicions de generació (C12). Aquests nodes han estat seleccionats arbitràriament, però en cap d'ells es viola la condició associada. Finalment, en el pas (16) es selecciona la condició C12, la qual està violada i cal reparar-la afegint l'esdeveniment δNumss(Mercè, 12345) a la transacció. Observi's que en aquest cas, aquesta reparació no pot violar cap de les condicions del Graf de Precedències Actiu.

Al final d'aquest pas, obtenim un Graf completament desmarcat, per tant, podem assegurar que la transacció obtinguda T'={δEdat(Mercè), δTreb(Mercè, Gu), δNumss(Mercè,12345)} és una solució a la petició inicial T={δEdat(Mercè)}.

□

6.4.4 Estimació de l'eficiència del Mòdul de Manteniment de la Consistència

El guany en eficiència aconseguit amb el mòdul de Manteniment de la Consistència el mesurem mitjançant un indicador que comptabilitza el nombre total de condicions que han estat comprovades per a generar una solució S. Amb aquest indicador es compten el nombre de condicions (diferents) que han estat comprovades i el nombre de vegades que es comprova una mateixa condició.

Aquest indicador serveix per a establir l'esforç necessari per a assegurar que una transacció T no viola cap restricció d'integritat i que satisfà realment la petició d'actualització U. Tenint en compte aquest indicador, podem establir que el mètode que asseguri la consistència d'una transacció T comprovant el menor nombre de condicions i el menor nombre de vegades cada una d'aquestes serà un mètode amb el millor nivell d'eficiència.

El lema següent estableix el nombre de condicions que caldrà comprovar per a assegurar que una transacció T és una solució, és a dir, que no viola cap de les condicions del Graf de Precedències Actiu que té associat.

Lema 6.1 Sigui N el número total de condicions associades a les restriccions d'integritat i al conjunt de condicions C. Considerem un Graf de Precedències Actiu que no contingui cicles. Llavors, el nombre de condicions que s'han de comprovar durant el procés de manteniment de la consistència per a obtenir una solució és de N.

Exemple 6.30: A l'exemple 6.29, s'han comprovat un total de 16 condicions. Cada condició ha estat comprovada un sol cop, excepte que la condició C5 que forma part d'un cicle i

ha estat comprovada 2 vegades. En el cas de no considerar el Graf de Precedències, i utilitzant un ordre arbitrari per a comprovar les mateixes condicions, s'haurien comprovat en mitjana 30 condicions[2].

□

Observi's que aquest indicador no es veu afectat pel nombre de reparacions que es realitzin en el procés de manteniment de les condicions del Graf de Precedències. És aquest el gran avantatge que comporta la utilització d'aquest Graf, ja que determina l'ordre en que cal comprovar aquestes condicions, tenint en compte les possibles reparacions que es puguin produir.

Per altra banda, cal observar que les condicions del conjunt de Condicions de Comprovació Cc es comprovaran diverses vegades, tantes com reparacions es realitzin durant el procés de manteniment de la consistència. De totes formes, cal valorar el fet que mantenint aquestes condicions en un conjunt apart del Graf de Precedències Actiu, permetrà que en certs casos s'eviti el comprovar un nombre indeterminat de condicions, que d'altra forma es comprovarien innecessàriament en cas de tenir que rebutjar una transacció T.

En el cas què en el Graf de Precedències Actiu existissin cicles en temps d'execució, el nombre de condicions total visitades s'hauria d'incrementar, per cada cicle, amb el nombre de condicions que formen el cicle multiplicat pel nombre de recorreguts que cal fer del cicle. Aquesta informació és difícil d'estimar a priori ja que depèn directament de la transacció i del contingut de la base de dades.

Una millora addicional que ja s'ha comentat en la secció 6.4.1 consisteix en la utilització d'un mètode per a comprovar si una condició és independent o no d'una actualització. En la generació del Graf de Precedències Actiu i després de cada reparació, aquest mètode pot ser especialment útil per a fer un marcatge dels nodes més intel·ligent. En aquest sentit, es marcarien únicament aquells nodes, les condicions dels quals, poden realment ser violades per la transacció T. Així doncs, es reduirien encara més el nombre de condicions que cal comprovar.

Cal dir de totes formes, que aquest no és l'únic indicador que es pot establir per a valorar el nivell d'eficiència del procés de manteniment de la consistència. Altres indicadors com el nombre d'accessos a la base de dades necessaris per a comprovar (i reparar) cada una de les condicions permetria fer una anàlisi més detallada de l'eficiència del mètode considerat. En aquest sentit, cal tenir en compte que les Condicions d'Actualització de Vistes que apareixen en el Graf de Precedències Actiu s'han obtingut d'un procés d'actualització de vistes, i per tant,

---

[2] Considerem la fórmula C+R(C/2) on C és el nombre total de condicions i R el nombre de reparacions realitzades.

són condicions que han estat especialitzades i no contenen referències a la base de dades extensional (veure secció 6.5.1).

6.5 Mòdul d'Actualització de Vistes

L'objectiu del mòdul d'Actualització de Vistes és, donada una petició d'actualització U, traduir aquesta petició en un conjunt d'actualitzacions de fets bàsics que satisfacin aquesta petició. Per a realitzar aquesta traducció, es té en compte l'eficiència tant del propi procés de traducció com la forma d'accedir a la informació requerida per a fer la traducció. En aquest sentit, aquest mòdul pretén reduir el nombre d'accessos que es fan a la base de dades extensional i pretén que únicament es considerin aquelles alternatives que condueixin a traduccions correctes.

Les peticions d'actualització U que rep aquest mòdul inclouen actualitzacions de fets derivats, i també poden incloure actualitzacions que cal evitar de fets bàsics i/o derivats. Aquests dos tipus d'actualitzacions es representen mitjançant esdeveniments derivats positius i mitjançant esdeveniments bàsics i derivats negats, respectivament. Les peticions d'actualització que sols inclouen actualitzacions bàsiques no requereixen un procés d'actualització de vistes ja que, en l'arquitectura proposada pel mètode, són processades directament pel mòdul de Manteniment de la Consistència.

Així doncs, donada una petició d'actualització U, aquest mòdul obté el conjunt de totes les traduccions a la petició d'actualització U. Cada una d'aquestes traduccions està composada per una transacció T, que inclou les actualitzacions de fets bàsics necessàries per a satisfer U, i el conjunt de condicions C, que ens asseguraran que la transacció T realment satisfà U. Un cop obtingudes totes aquestes traduccions, es trametran al Mòdul de Manteniment de la Consistència per tal de què comprovi la seva consistència.

Per tal de millorar l'eficiència en el procés de traducció d'una vista, s'ha dividit aquest procés en dues etapes: una primera etapa d'Anàlisi i una etapa posterior de Traducció. En la primera etapa, s'analitza la petició d'actualització i s'hi realitzen tots els accessos a la base de dades extensional que són necessaris per a traduir aquesta petició. A més a més, es determinen quines són les regles de la base de dades augmentada que permeten obtenir una traducció. A l'etapa de Traducció, és on realment s'obtenen les traduccions a la petició d'actualització amb la construcció dels conjunts T i C.

L'eficiència en aquest procés d'actualització de vistes s'assoleix amb una reducció dels accessos a realitzar a la base de dades extensional, ja que tots ells es realitzen de forma avançada en l'etapa d'Anàlisi, sense repetir accessos i eliminant accessos redundants. Per altra banda, la informació obtinguda a partir de l'anàlisi de la petició d'actualització i del contingut de la base de dades permet identificar i descartar aquelles regles de la base de dades augmentada que no

permeten obtenir cap traducció vàlida de la petició d'actualització i, sols considerar aquelles que si permeten obtenir una traducció vàlida.

Les etapes d'Anàlisi i Traducció estan descrites respectivament a les seccions 6.5.1 i 6.5.2. En cada una d'aquestes seccions es descriuen en detall les tècniques utilitzades per a millorar l'eficiència del procés de traducció d'una petició d'actualització.

6.5.1. Etapa d'Anàlisi

L'objectiu de l'etapa d'Anàlisi és el de realitzar un treball preparatori per tal de millorar l'eficiència del procés de traducció, el qual es realitzarà en l'etapa de Traducció. Aquesta etapa es correspon al Mòdul d'Anàlisi de l'arquitectura general del mètode presentat a la Fig. 6.2.

A l'etapa d'Anàlisi es diferencien clarament dues activitats: en primer lloc, a partir de la petició d'actualització U, es determinen i realitzen les consultes a la base de dades que són necessàries per a traduir la petició U. El resultat d'aquestes consultes és el que anomenem extensió rellevant de la petició d'actualització. En segon lloc, es determinen quines regles d'esdeveniment de la base de dades augmentada A(D) permeten generar alguna traducció a la petició d'actualització i s'especialitzen respecte a l'extensió obtinguda amb les consultes anteriors.

Al final d'aquesta etapa s'incorporen a la base de dades augmentada A(D) les regles d'esdeveniment que han estat especialitzades. Així, a l'etapa de Traducció, al fer el Manteniment de la Consistència, i a posteriors crides al Mòdul d'Actualització de Vistes es podrà aprofitar aquesta especialització de les regles d'esdeveniment. L'extensió rellevant calculada en aquesta etapa també es recordarà per a posteriors etapes del mètode per tal d'evitar que es repeteixin accessos a la base de dades extensional.

Determinació de l'extensió rellevant a la petició d'actualització

Les regles d'esdeveniment associades a un predicat derivat són les que defineixen les diferents formes com es pot induir una actualització d'aquest predicat derivat. Així doncs, per a traduir una petició d'actualització d'un predicat derivat s'utilitzaran les seves regles d'esdeveniment.

La definició de cada una d'aquestes regles determina l'estat de la base de dades en que aquesta regla pot generar una traducció. Els literals del cos d'aquestes regles que es refereixen a predicats de la base de dades estableixen els requeriments sobre els fets de la base de dades que cal que es satisfacin per a obtenir una traducció a la petició d'actualització del predicat derivat associat.

Exemple 6.31: Consideri's el predicat derivat Actiu(p) i les regles d'esdeveniment següents:

(I9) $\iota Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \neg\delta Emp(\underline{p},c) \wedge \neg Aux3(\underline{p},c) \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p})$

(I10) $\iota Actiu(\underline{p}) \leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Baixa(\underline{p}) \wedge \neg\iota Baixa(\underline{p})$

(I11) $\iota Actiu(\underline{p}) \leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p})$

(I12) $\iota Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \mu Emp(\underline{p},c,c1) \wedge c \neq c1 \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p})$

(D5) $\delta Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \delta Emp(\underline{p},c) \wedge \neg Baixa(\underline{p})$

(D6) $\delta Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \neg Baixa(\underline{p}) \wedge \iota Baixa(\underline{p})$

$Aux11(\underline{p}) \leftarrow Emp(\underline{p},c)$

$Aux3(\underline{p},c) \leftarrow \mu Emp(\underline{p},c,c1)$

Observi's que en aquest exemple, les regles d'esdeveniment d'esborrat són totes aplicables a un estat de la base de dades en que el fet Emp(p,c) és cert i el fet Baixa(p) és fals, és dir, quan el fet Actiu(p) és cert. Per altra banda, existeix almenys una regla d'esdeveniment d'inserció per a cada possible estat de la base de dades en que el fet Actiu(p) és fals. És a dir, les regles I9 i I12 són aplicable quan els dos fets Emp(p,c) i Baixa(p) són certs; les regla I10 i I11 són aplicables quan $\neg\exists c$ Emp(p,c) que sigui cert a la base de dades, però I10 requereix que Baixa(p) sigui fals, mentre que I11 requereix que sigui cert. Així doncs, les regles d'esdeveniment són aplicables a diferents estats de la base de dades, mentre que les d'esborrat són totes aplicables al mateix estat.

□

Per a determinar les regles d'esdeveniment que permeten obtenir una traducció de vista, cal conèixer quin és el contingut de base de dades. De totes formes, per a traduir una petició concreta d'actualització d'un predicat derivat, no cal conèixer tot el contingut de la base de dades extensional, sinó que solament cal tenir en compte fets bàsics que defineixen la vista a actualitzar. Així doncs, podem definir que l'extensió rellevant d'una petició d'actualització d'un fet derivat es correspon únicament a l'extensió dels fets bàsics que defineixen directament o indirectament el fet derivat a actualitzar.

Definició 6.19. Sigui Ev(k, x) un esdeveniment associat a un predicat P, el qual està definit directament o indirectament en termes dels predicats bàsics $Q_i(\underline{h_i}, y_i)$. Aleshores, els *predicats rellevants a l'esdeveniment Ev* són els predicats bàsics $Q_i(\underline{h_i}, y_i)\theta$, on θ es correspon a la unificació $(k/h_i, x/y_i)$.

Exemple 6.32: Els predicats rellevants a l'esdeveniment ιActiu(p) són el conjunt de predicats {Treb(p,c), Cont(p,c), Baixa(p)}.

□

De forma similar, i tenint en compte quins d'aquests predicats rellevants són certs a la base de dades extensional, poder definir quina és l'extensió rellevant d'un esdeveniment derivat.

Definició 6.20. Sigui PR={$Q_i(\underline{h_i},y_i)\theta$} el conjunt de predicats rellevants a un esdeveniment derivat Ev, aleshores, *l'extensió rellevant* de *l'esdeveniment Ev* és l'extensió de cada un dels predicats bàsics $Q_i(\underline{h_i}, y_i)\theta$.

Exemple 6.33: L'extensió rellevant de l'esdeveniment ιActiu(p) de l'exemple 6.32 es correspon al conjunt de fets bàsics següents:

(F1) Treb(Núria, Za)
(F2) Cont(Núria, Za)
(F4) Treb(Sílvia, SJD)
(F5) Treb(Toni, AIC)
(F6) Cont(Toni, AIC)
(F8) Baixa(Toni)
(F9) Treb(Mercè, EDM)

□

Generalitzant aquestes definicions per a més d'un esdeveniment, podem definir l'extensió rellevant d'una petició d'actualització U qualsevol.

Definició 6.21. L'*extensió rellevant d'una petició d'actualització U* es correspon a la unió de les extensions rellevants de cada un dels esdeveniments de la petició U.

És fàcilment comprovable, que els fets bàsics de l'extensió rellevant d'una petició d'actualització U són els únics fets de la base de dades extensional que cal consultar per tal de traduir la petició U. Per tal d'obtenir totes les traduccions a una petició U, únicament serà necessari conèixer la certesa o falsedat de cada un d'aquests fets i cap altre fet bàsic de la base de dades serà consultat durant tot el procés de traducció.

Així doncs, els predicats rellevants dels esdeveniments de la petició d'actualització U determinen quines consultes cal realitzar a la base de dades extensional. Per a cada predicat bàsic rellevant es realitza una consulta a la base de dades. Amb el resultat de totes aquestes consultes (l'extensió rellevant) i les regles de derivació es podran resoldre totes les referències a predicats de la base de dades que apareguin a les regles d'esdeveniment associades als esdeveniments de la petició d'actualització U.

Exemple 6.34: Donada la petició d'actualització U = {ιActiu(Toni)}, l'extensió rellevant d'aquesta petició U és el conjunt de fets bàsics {Treb(Toni, AIC), Cont(Toni, AIC), Baixa(Toni)}.

□

Observi's que amb el càlcul de l'extensió rellevant a la petició d'actualització U, hem realitzat de forma avançada tots els accessos que seran necessaris per a traduir la petició U. Així doncs, en tot el mòdul d'Actualització de Vistes no caldrà realitzar cap altre accés a la base de dades extensional.

A més a més, amb la determinació dels predicats rellevants als esdeveniments de la petició U, hem establert el nombre de consultes que cal realitzar a la base de dades. Una anàlisi més precisa dels accessos a la base de dades que són necessaris per a calcular l'extensió rellevant es realitza a la secció 6.5.3.

Així doncs, la primera millora a nivell d'eficiència, que s'assoleix amb el treball preparatori realitzat en aquesta etapa d'Anàlisi, és una reducció del nombre d'accessos a la base de dades extensional necessaris durant tot el procés d'actualització d'una vista.

Especialització de les regles d'esdeveniment

Una vegada es té coneixement de quins fets són certs a la base de dades i quins són falsos, es poden identificar quines són les regles d'esdeveniment que donaran lloc a una traducció de la petició U. Al mateix temps es podran identificar aquelles que no poden generar cap traducció i es podran descartar del procés de traducció.

En aquest pas de l'etapa d'Anàlisi i per a distingir entre les regles d'esdeveniment que poden generar una traducció i les que no, proposem l'especialització de les regles d'esdeveniment involucrades directament o indirectament amb la petició d'actualització U. Aquesta especialització es realitza tenint en compte l'extensió rellevant de la petició U obtinguda en el pas anterior.

L'especialització de les regles d'esdeveniment es realitza adaptant la transformació proposada a [KS90] en el marc de la gestió d'excepcions a regles de la programació lògica. En el nostre cas, aquesta transformació és utilitzada per a convertir les regles d'esdeveniment de la base de dades augmentada A(D) en dos regles d'esdeveniment especialitzades: una regla específica que és aplicable a una instància concreta t' i una regla general aplicable a la resta d'instàncies $t \neq t'$.

Definició 6.22. Sigui P(t') un esdeveniment derivat instanciat, on t' és un vector de constants. Sigui E l'extensió rellevant de l'esdeveniment P(t') i R una regla d'esdeveniment $P(t) \leftarrow L_1 \wedge ... \wedge L_n$, on t és un vector de termes. Les *Regles d'Esdeveniment Especialitzades R'* de P(t') es defineixen de la forma següent:

(R_1') $P(t) \leftarrow L_1 \wedge ... \wedge L_n \wedge t \neq t'$

(R_2') $[P(t) \leftarrow L_1 \wedge ... \wedge L_n]\theta$

on θ es correspon a l'unificador més general entre t i t', i on els literals de la regla R_2' que fan referència a predicats de la base de dades han estan avaluats respecte a l'extensió rellevant E.

Exemple 6.35: Siguin les regles d'esdeveniment I9 i I10 de l'exemple 6.31 i l'extensió rellevant de l'exemple 6.34. Aleshores la regles especialitzades obtingudes són les següents:

$(I9_1')$ $\iota Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \neg\delta Emp(\underline{p},c) \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p}) \wedge p \neq Toni$

$(I9_2')$ $\iota Actiu(\underline{Toni}) \leftarrow \neg\delta Emp(\underline{Toni},AIC) \wedge \delta Baixa(\underline{Toni})$

$(I10_1')$ $\iota Actiu(\underline{p}) \leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Baixa(\underline{p}) \wedge \neg\iota Baixa(\underline{p}) \wedge p \neq Toni$

□

Observi's que en el cas de que t i t' no es puguin unificar, la regla especialitzada obtinguda R' no modifica la regla d'esdeveniment R de la base de dades augmentada (R=R'). De forma similar, en el cas de que algun literal de la regla d'esdeveniment R que fa referència a un fet de la base de dades no es satisfaci, llavors la regla especialitzada obtinguda es correspon a $R'=R_1'$. Altrament, amb aquesta especialització, s'obtenen dues regles d'esdeveniment especialitzades R_1' i R_2' que substitueixen a la regla R en la base de dades augmentada A(D).

Per a obtenir una especialització completa d'una regla R en la que apareixen, en el seu cos, esdeveniments derivats, cal aplicar la mateixa transformació a les regles d'esdeveniment dels esdeveniments que apareixen al cos de la regla especialitzada R_2'.

En el cas de que una regla ja especialitzada per a una instància t' calgui especialitzar-la de nou per a una nova instància t'' (t'≠t''), aplicarem la transformació anterior al resultat de la primera especialització R'. Observi's que en aquest casos, solament s'especialitzarà la regla R_1', ja que R_2' no s'unificarà amb la nova instància t''.

Exemple 6.36: Donada la petició d'actualització de l'exemple 6.34 i l'extensió rellevant d'aquesta petició, el conjunt de totes les regles d'esdeveniment especialitzades associades a la petició U és el següent:

$(I9_1')$ $\iota Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \neg\delta Emp(\underline{p},c) \wedge \neg Aux3(\underline{p},c) \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p}) \wedge p \neq Toni$

$(I9_2')$ $\iota Actiu(\underline{Toni}) \leftarrow \neg\delta Emp(\underline{Toni},AIC) \wedge \neg Aux3(\underline{Toni},AIC) \wedge \delta Baixa(\underline{Toni})$

$(I10_1')$ $\iota Actiu(\underline{p}) \leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge \neg Baixa(\underline{p}) \wedge \neg\iota Baixa(\underline{p}) \wedge p \neq Toni$

$(I11_1')$ $\iota Actiu(\underline{p}) \leftarrow \neg Aux11(\underline{p}) \wedge \iota Emp(\underline{p},c) \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p}) \wedge p \neq Toni$

$(I12_1')$ $\iota Actiu(\underline{p}) \leftarrow Emp(\underline{p},c) \wedge \mu Emp(\underline{p},c,c1) \wedge c \neq c1 \wedge Baixa(\underline{p}) \wedge \delta Baixa(\underline{p}) \wedge p \neq Toni$

$(I12_2')$ $\iota Actiu(\underline{p}) \leftarrow \mu Emp(\underline{Toni},AIC,c1) \wedge AIC \neq c1 \wedge \delta Baixa(\underline{Toni})$

$(D1_1')$ $\delta Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge \delta Treb(\underline{p},c) \wedge Cont(\underline{p},c) \wedge p \neq Toni$

$(D1_2')$ $\delta Emp(\underline{Toni},AIC) \leftarrow \delta Treb(\underline{Toni},AIC)$

$(D2_1')$ $\delta Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c1) \wedge c1 \neq c \wedge Cont(\underline{p},c) \wedge \neg\mu Cont(\underline{p},c,c1) \wedge p \neq Toni$

$(D2_2')$ $\delta Emp(\underline{Toni},AIC) \leftarrow \mu Treb(\underline{Toni},AIC,c1) \wedge c1 \neq AIC \wedge \neg\mu Cont(\underline{Toni},AIC,c1)$

$(D3_1')$ $\delta Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge Cont(\underline{p},c) \wedge \delta Cont(\underline{p},c) \wedge p \neq Toni$

$(D3_2')$ $\delta Emp(\underline{Toni},AIC) \leftarrow \delta Cont(\underline{Toni},AIC)$

$(D4_1')$ $\delta Emp(\underline{p},c) \leftarrow Treb(\underline{p},c) \wedge Cont(\underline{p},c) \wedge \mu Cont(\underline{p},c,c1) \wedge c1 \neq c \wedge \neg\mu Treb(\underline{p},c,c1) \wedge p \neq Toni$

$(D4_2')$ $\delta Emp(\underline{Toni},AIC) \leftarrow \mu Cont(\underline{Toni},AIC,c1) \wedge c1 \neq AIC \wedge \neg\mu Treb(\underline{Toni},AIC,c1)$

$(M1_1')$ $\mu Emp(\underline{p},c,c1) \leftarrow Treb(\underline{p},c) \wedge \mu Treb(\underline{p},c,c1) \wedge Cont(\underline{p},c) \wedge \mu Cont(\underline{p},c,c1) \wedge c \neq c1 \wedge p \neq Toni \wedge c \neq AIC$

$(M1_2')$ $\mu Emp(\underline{Toni},AIC,c1) \leftarrow \mu Treb(\underline{Toni},AIC,c1) \wedge \mu Cont(\underline{Toni},AIC,c1) \wedge AIC \neq c1$

$(A3_1')$ $Aux3(\underline{p},c) \leftarrow \mu Emp(\underline{p},c,c1) \wedge p \neq Toni \wedge c \neq AIC$

$(A3_2')$ $Aux3(\underline{Toni},AIC) \leftarrow \mu Emp(\underline{Toni},AIC,c1)$

□

De les dues regles especialitzades (R_1' i R_2') que es poden obtenir a partir de la regla d'esdeveniment R, la regla R_2' és específica per a la instància t' obtinguda de l'esdeveniment derivat que es vol traduir, per tant, serà aquesta regla la que en l'etapa de Traducció ens permetrà obtenir una traducció a aquest esdeveniment derivat. Per altra banda, la regla R_1' s'incorpora a la base de dades augmentada amb la finalitat de no perdre completesa, i el terme $t \neq t'$ permetrà, que en l'etapa de Traducció, no es consideri aquesta regla.

Aleshores, podem observar que amb aquest procediment d'especialització de les regles d'esdeveniment respecte a la petició d'actualització U i la seva extensió rellevant, hem identificat (i parcialment avaluat) les regles que poden conduir a traduccions de U i, a la vegada, hem descartat aquelles que no conduiran a cap traducció vàlida. Aquesta distinció permet reduir considerablement el nombre de regles a tenir en compte en la traducció d'una petició d'actualització U millorant d'aquesta forma l'eficiència del propi procés de traducció.

Les dues regles especialitzades són equivalents a la regla inicial R i, per tant, poden substituir-la en la base de dades augmentada A(D) sense perdre completesa. Per tant, es substitueix cada una de les regles d'esdeveniment per la seva especialització. Aleshores, per a obtenir una base de dades augmentada especialitzada respecte a la petició d'actualització U, caldrà aplicar aquesta transformació a totes les regles d'esdeveniment associades a cada un dels esdeveniments derivats (positius i negatius) que apareixen a la petició d'actualització U.

Mentre no es modifiqui la base de dades extensional, les regles especialitzades segueixen essent vàlides i, per tant, durant tot el procés de traducció d'una petició d'actualització i durant el procés de manteniment de la consistència d'aquestes traduccions, es podran seguir utilitzant les regles d'esdeveniment especialitzades de la mateixa manera que s'utilitzarien les regles d'esdeveniment no especialitzades.

A continuació es presenta l'algorisme encarregat d'especialitzar la base de dades augmentada A(D) a partir d'una petició d'actualització U i l'extensió rellevant associada E:

```
Procediment Especialitzar_Base_de_Dades (A(D), U, Ext)
    /* entrada: base de dades augmentada A(D), conjunt d'esdeveniments U i extensió rellevant Ext */
    /* sortida: base de dades augmentada A(D) amb regles especialitzades */

per cada Ev ∈ U fer
    si Ev és esdeveniment derivat llavors
        per cada regla d'esdeveniment R que defineix Ev fer
            R' := Especialitzar_Regla(R, Ev, Ext);          /* R' = [R1', R2']*/
            Substituir(R, R', A(D));
            DEv := Obtenir_esdeveniments_derivats(R2');
            U := U ∪ DEv
        fi per cada;
    fi si;
fi per cada;
fi procediment;
```

La funció Especialitzar_Regla(R, Ev, Ext) correspon a la transformació de la definició 6.20. Observi's que aquesta funció s'aplica a tots els esdeveniments que apareixen a la petició U i als esdeveniments derivats que apareixen a les regles especialitzades R_2'. El resultat del procediment Especialitzar_Base_de_Dades és la base de dades augmentada A(D) on s'han substituït algunes regles d'esdeveniment per la seva especialització.

Al final de l'etapa d'Anàlisi hem obtingut una base de dades augmentada A(D) on les regles d'esdeveniment associades a la petició d'actualització U han estat especialitzades respecte a l'extensió rellevant a aquesta petició d'actualització. Aquest treball preparatori realitzat en aquesta etapa és especialment útil per a l'etapa de Traducció, ja que per a generar totes les traduccions a la petició U, solament es consideren les regles d'esdeveniment especialitzades R_2' i, a més a més, no es requeriran nous accessos a la base de dades extensional, ja que tots els accessos s'han realitzat en aquesta etapa al calcular l'extensió rellevant a U. Per altra banda, mentre no s'actualitzi la base de dades extensional, es podran seguir utilitzant les regles especialitzades i l'extensió rellevant obtinguda, millorant l'eficiència en el procés de manteniment de la consistència.

6.5.2. Etapa de Traducció

L'objectiu d'aquesta etapa és, donada una petició d'actualització U i la base de dades augmentada A(D), obtenir totes les formes d'actualitzar la base de dades extensional que permetin satisfer la petició d'actualització U.

El resultat que s'obté al final d'aquesta etapa és el conjunt de totes les traduccions a la petició d'actualització U. Cada una d'aquestes traduccions està composada per una transacció T i un conjunt de condicions C. La transacció T conté les actualitzacions de fets bàsics necessàries per a satisfer la petició U i el conjunt C conté les condicions que ens asseguren que la transacció T realment satisfà la petició d'actualització U.

Concretament, aquesta etapa té com a propòsit el de construir els conjunts T i C a partir de les regles d'esdeveniments especialitzades de l'A(D). Per a construir-los, donada la petició

d'actualització U, s'aplicarà un procediment de desplegament sobre els esdeveniments derivats positius de U fins a obtenir una forma normal disjuntiva en que tots els esdeveniments positius siguin esdeveniments bàsics. Aleshores, per a cada disjunctand es genera una parella de conjunts T i C, on els esdeveniments positius s'afegeixen al conjunt T i els literals negats s'afegeixen al conjunt C com a condicions. Els esdeveniments bàsics negats s'afegeixen directament al conjunt C, i pels esdeveniments derivats negats es consideraran les seves regles d'esdeveniment com a diferents condicions.

La funció Tradueix_Petició és l'encarregada de realitzar aquest procés. A continuació es presenta l'algorisme d'aquesta funció:

```
Funció Tradueix_Petició (U, A(D)): conjunt de parells (T, C);
    /* entrada: una petició d'actualització U i la base de dades augmentada A(D)*/
    /* sortida: conjunt de totes les traduccions TC = { (T, C) } */

Desplegament(U, A(D), FND);
per cada Disj de DNF fer
    T := ∅; C := ∅;
    per cada Ev ∈ Disj fer
        si positiu(Ev) llavorsT := T ∪ {Ev};
        sino
            si esdeveniment_bàsic(Ev) llavors    C := C ∪{← Ev};
            sino
                Obtenir_regles_esdeveniment(Ev, A(D), EvR);
                per cada Ci ∈ EvR fer
                    C := C ∪ {← Ci };
                fi per cada;
            fi si;
        fi si;
    fi per cada;
    TC := TC ∪ { (T, C) };
fi per cada;
retornar(TC);
```

Observi's que la regla especialitzada R_2' és la regla d'esdeveniment que permet obtenir una traducció a la petició d'actualització U, ja que es correspon a l'especialització per a la instància t' de l'esdeveniment Ev(t') de la petició U. La regla R_1' no es considerarà ja que ho impedeix el literal t'≠t. A més a més, observi's que en la generació dels conjunts T i C, no cal fer cap accés a la base de dades extensional, ja que a la regla R_2' no té literals que facin referència a predicats de la base de dades ja que han estat avaluats durant l'especialització de la regla a l'etapa d'Anàlisi.

Exemple 6.37: Donades la petició d'actualització U i l'extensió rellevant d'aquesta petició de l'exemple 6.34 i les regles d'esdeveniment especialitzades obtingudes a l'exemple 6.36, aleshores, la única traducció TC que s'obté per aquesta petició, un cop aplicada la funció Tradueix_Petició és la següent:

$T_1 = \{\delta Baixa(\underline{Toni})\}$

$C_1 = \{\leftarrow \delta Treb(\underline{Toni},AIC);$

$\leftarrow \mu Treb(\underline{Toni},AIC,c1) \wedge c1 \neq AIC \wedge \neg\mu Cont(\underline{Toni},AIC,c1);$

$\leftarrow \delta Cont(\underline{Toni},AIC);$

$\leftarrow \mu Cont(\underline{Toni},AIC,c1) \wedge c1 \neq AIC \wedge \neg\mu Treb(\underline{Toni},AIC,c1)$

$\leftarrow \mu Treb(\underline{Toni},AIC,c1) \wedge \mu Cont(\underline{Toni},AIC,c1) \wedge AIC \neq c1\}$

□

Un cop finalitzada l'etapa de Traducció s'obtenen totes les traduccions a la petició U, representades per les parelles de conjunts T i C. Cal observar que es pot donar el cas que alguna de les traduccions obtingudes no sigui correcte. Per una banda, es pot donar el cas que la transacció T no satisfaci realment la petició U degut a que viola alguna de les condicions del conjunt C. Per altra banda, es pot donar el cas de que encara que la transacció T satisfaci les condicions de C, aquesta violi alguna de les restriccions d'integritat de la base de dades. La comprovació de la consistència de la transacció T respecte al conjunt C i a les restriccions d'integritat es realitzarà al final del procés d'Actualització de Vistes al executar-se el mòdul de Manteniment de la Consistència. Per tant, no podrem assegurar que una traducció és correcte fins que s'hagi assegurat la seva consistència amb el mòdul de Manteniment de la Consistència.

6.5.3. Estimació de l'eficiència del Mòdul d'Actualització de Vistes

El guany en eficiència aconseguit amb el mòdul d'Actualització de Vistes el mesurem mitjançant dos indicadors. Per una banda estimem quin és el nombre d'accessos que cal realitzar a la base de dades extensional per tal de traduir una petició d'actualització U. Per altra banda, estimem quantes regles d'esdeveniment de la base de dades augmentada A(D) són realment útils per a generar una traducció d'aquesta petició.

L'estimació d'accessos a la base de dades extensional és un indicador inherent a l'etapa d'Anàlisi d'aquest mòdul, ja que és l'únic lloc en aquest mòdul on es considera el contingut de la base de dades extensional. En canvi, l'estimació de regles d'esdeveniment que condueixen a una traducció és un indicador més relacionat amb l'etapa de Traducció, ja que estima l'esforç necessari per a obtenir les traduccions en aquesta etapa.

Aquests dos indicadors han estat escollits perquè, el primer d'ells, és un dels indicadors més comuns i utilitzats per avaluar el cost de tot procés que requereix d'una base de dades per emmagatzemar-hi informació. En concret i en el nostre cas, aquest indicador ens dóna una idea del grau d'eficiència a l'hora de cercar la informació necessària per a fer la traducció d'una vista. El segon d'ells ha estat escollit per avaluar l'eficiència del propi procés de traducció de la vista. Aquest indicador, estima l'espai de recerca que cal considerar per tal d'obtenir les

traduccions a una vista. En el nostre cas, aquest es pot considerar un indicador indirecte del cost computacional del propi procés de traducció.

Cal tenir en compte que per a realitzar l'estimació d'aquests indicadors s'ha considerat un cas restringit, encara que es comentarà com estendre els resultats a casos més complexes. El cas que s'ha considerat és el següent:

- El predicat derivat P que es vol actualitzar està definit per la següent regla de derivació:

$$P(x) \leftarrow L1 \wedge ... \wedge Ln \qquad (1)$$

on cada literal Li es correspon a un predicat bàsic i on tota variable que apareix en algun literal Li també apareix al cap de la regla.

- La clau del predicat P està composada per tots els arguments d'aquest predicat, per tant, solament es consideren esdeveniments d'inserció i d'esborrat.

El cas considerat ha estat restringit degut a que, per una banda, aquests indicadors es faran servir per a comparar amb altres treballs en l'àmbit de l'actualització de vistes, els quals no consideren la modificació com un operador elemental d'actualització, i per altra banda, perquè l'estimació per a casos més complexes és fàcilment extensible a partir d'aquest cas restringit.

A continuació analitzarem amb més detall aquests indicadors i establirem una estimació global dels mateixos per al nostre mètode. Diferenciem el cas de la inserció d'un fet del predicat P i del cas de l'esborrat d'un fet del mateix predicat.

Inserció d'un fet derivat $\iota P(\underline{x})$

En el cas de la inserció d'un fet derivat, la definició d'aquest esdeveniment estableix que per a poder realitzar-se, cal assegurar que el fet $P(\underline{x})$ no és cert a la base de dades, és a dir, que no es pot deduir a partir de les regles de derivació que defineixen P i dels fets emmagatzemats a la base de dades extensional.

Tenint en compte la definició (1) del predicat $P(\underline{x})$, un fet d'aquest predicat serà fals quan un (o més d'un) dels literals Li del cos de la regla no es satisfacin a la base de dades. Així doncs, hi ha 2^n-1 formes diferents de fer fals aquest predicat, i per tant, per a cada un d'aquests casos, la inserció del fet $P(\underline{x})$ s'assolirà actualitzant de forma diferent la base de dades extensional. Cada un d'aquests casos són els que contemplen cada una de les regles d'esdeveniment d'inserció $\iota P(\underline{x})$ de la base de dades augmentada A(D).

Així doncs, donada una base de dades extensional i una petició d'inserció d'un fet $P(\underline{x})$, per a identificar, quina de les 2^n-1 regles d'esdeveniment d'inserció es pot aplicar, tenint en compte el contingut de la base de dades actual, cal identificar prèviament quina de les 2^n-1 formes de fer fals el fet $P(\underline{x})$ és satisfeta per la base de dades extensional.

En el nostre mètode, utilitzem el concepte de predicat rellevant i solament necessitem realitzar n accessos a la base de dades extensional: un accés per a cada fet bàsic associat a cada literal Li i, a partir del resultat d'aquestes consultes identifiquem quina regla d'esdeveniment d'inserció es pot utilitzar per a traduir la inserció del fet derivat P(x). Observi's a més a més, que amb el procés d'especialització de les regles d'esdeveniment, aconseguim que la resta de regles d'inserció no es considerin en la resta del procés de traducció.

Exemple 6.38: Consideri's la petició d'actualització U={ιP(1,2)} i la base de dades augmentada següent:

Q(3,4)

$P(x,y) \leftarrow Q(x,y) \wedge \neg R(y)$

$\iota P(x,y) \leftarrow Q(x,y) \wedge \neg \delta Q(x,y) \wedge R(y) \wedge \delta R(y)$

$\iota P(x,y) \leftarrow \neg Q(x,y) \wedge \iota Q(x,y) \wedge \neg R(y) \wedge \neg \iota R(y)$

$\iota P(x,y) \leftarrow \neg Q(x,y) \wedge \iota Q(x,y) \wedge R(y) \wedge \delta R(y)$

$\delta P(x,y) \leftarrow Q(x,y) \wedge \delta Q(x,y) \wedge \neg R(y)$

$\delta P(x,y) \leftarrow Q(x,y) \wedge \neg R(y) \wedge \iota R(y)$

Per a obtenir una traducció composada per T = { ιP(1,2)} i C = {←ιR(2)} ha calgut fer 2 accessos a la base de dades (Q(1,2)? i R(2)?) per a calcular l'extensió rellevant E = {¬Q(1,2), ¬R(2)} i s'ha considerat una única regla especialitzada: ιP(1,2) ← ιQ(1,2) ∧ ¬ιR(2). Observi's que en aquests cas es realitzen tants accessos a la base de dades extensional com predicats bàsics defineixen P(x,y) (n=2), i sols es considera una regla d'esdeveniment en la generació de la traducció.

□

Esborrat d'un fet derivat δP(x)

En el cas de l'esborrat d'un fet derivat, la definició d'aquest esdeveniment estableix que per a poder realitzar-se cal assegurar que el fet P(x) és cert a la base de dades.

En aquest cas, sols hi ha una forma d'assegurar que aquest fet és cert, i és precisament l'alternativa no considerada en el cas de la inserció, és a dir, quan tots els literals L_i del cos de la regla (1) es satisfan. Per altra banda, per a esborrar un fet derivat, tant sols cal que actualitzem la base de dades extensional de forma que un dels literals Li esdevingui fals, per tant, existeixen n formes diferents d'assolir aquest esborrat i especificades en n regles d'esdeveniment d'esborrat a la base de dades augmentada A(D).

Així doncs, donada una base de dades extensional i una petició d'esborrat d'un fet P(x), per a comprovar si el fet P(x) és cert, solament cal fer la consulta P(x)?, el qual comporta un sol accés lògic a la base de dades. Per altra banda, podem comprovar que totes les regles d'esdeveniment d'esborrat seran aplicables i permetran obtenir les n formes alternatives d'esborrar el fet P(x).

Exemple 6.39: Tenint en compte la mateixa base de dades augmentada de l'exemple 6.38, considerem en aquest cas la petició d'actualització U={δP(3,4)}. Per a obtenir l'extensió rellevant associada a la petició d'actualització, ha calgut un únic accés lògic a la base de dades, el qual es correspon a la consulta P(3,4)?. Les regles d'esdeveniment especialitzades que permeten obtenir una traducció són les següents:

$\delta P(3,4) \leftarrow \delta Q(3,4)$

$\delta P(3,4) \leftarrow \iota R(4)$

Aleshores, en aquest exemple obtindrem dues traduccions alternatives:

$T_1=\{\delta Q(3,4)\}$ amb $C_1=\varnothing$

$T_2=\{\iota R(4)\}$ amb $C_2=\varnothing$

Observi's que en aquest cas, per a obtenir totes les traduccions a la petició U, hem realitzat un únic accés lògic a la base de dades i hem considerat 2 regles d'esdeveniment. □

Un cop analitzats aquests dos casos, podem establir dos lemes que ens permeten quantificar, en part, l'eficiència del mètode proposat.

Lema 6.2: Sigui P un predicat derivat definit per una única regla $P(t_1, \ldots, t_m) \leftarrow L_1 \wedge \ldots \wedge L_n$, amb $n \geq 1$, on els literals $L_1,\ldots,L_n$ es corresponen a predicat bàsics i on tota variable que apareix als literals $L_1,\ldots,L_n$ també apareix als termes $t_1, \ldots, t_m$. Aleshores, el nombre d'accessos lògics a la base de dades extensional necessaris per a traduir una petició d'actualització de P és el següent:

- Una inserció $\iota P(k_1, \ldots, k_m)$: n accessos a la EDB
- Un esborrat $\delta P(k_1, \ldots, k_m)$: 1 accés a la EDB.

Lema 6.3: Sigui P un predicat derivat definit per una única regla $P(t_1, \ldots, t_m) \leftarrow L_1 \wedge \ldots \wedge L_n$, amb $n \geq 1$, on els literals $L_1,\ldots,L_n$ es corresponen a predicat bàsics i on tota variable que apareix als literals $L_1,\ldots,L_n$ també apareix als termes $t_1, \ldots, t_m$. Aleshores, el nombre de regles d'esdeveniments que es consideren en el procés de traducció d'una actualització de P és el següent:

- Una inserció $\iota P(k_1, \ldots, k_m)$: 1 regla d'esdeveniment d'inserció.
- Un esborrat $\delta P(k_1, \ldots, k_m)$: n regles d'esdeveniment d'esborrat.

Altres casos

Aquests mateixos lemes poden estendre's a definicions de predicats derivats més complexes:

- En el cas de tenir un predicat derivat definit per altres predicats derivats, cal aplicar els lemes anteriors a cada un dels predicats derivats que el defineixen.

  En el cas de la petició U={ιActiu(Toni)}, de l'exemple 3.37, els accessos realitzats han estat 3 i les regles considerades en la traducció 1. Cal observar que en aquest cas, s'han

especialitzat les regles de l'esdeveniment δEmp(Toni,AIC) i es tindran en compte durant el procés de manteniment de la consistència.

- De forma similar, si un predicat derivat està definit per vàries regles de derivació, cal aplicar els lemes anteriors a cada una de les regles que el defineixen.
- En el cas de tenir claus existencials, és a dir, en el cas en que algun argument de la clau d'algun predicat del cos de la regla no forma part de la clau del predicat derivat, aleshores, els mateixos lemes també poden aplicar-se directament. De totes formes, cal dir que, pel cas de la inserció, per a resoldre les referències a la base de dades que apareixen en les regles d'esdeveniment, es poden definir estratègies d'accés més eficients.
- En el cas de tenir un predicat derivat amb arguments clau i arguments no clau , aleshores és necessari el considerar l'esdeveniment de modificació. En aquest cas, aquests lemes són directament extensible per a poder considerar les regles d'esdeveniment de modificació i el nombre de regles d'esdeveniment de cada tipus.