

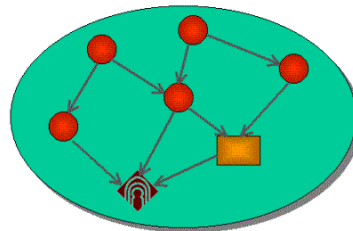
Compilación incremental de redes Bayesianas.- Un enfoque basado en la descomposición en subgrafos primos maximales

M. Julia Flores¹, José A. Gámez¹ y Kristian G. Olesen²

¹Univ. de Castilla-La Mancha

² Universidad de Aalborg

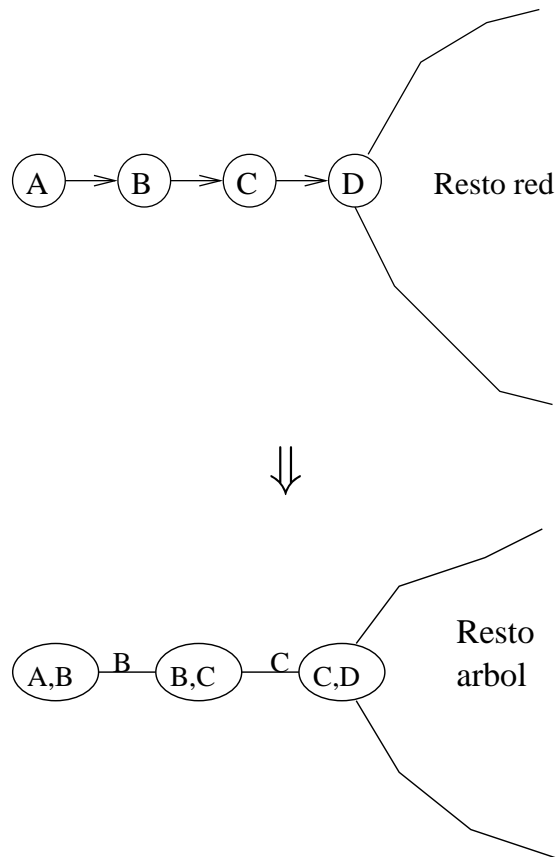
ISL
Intelligent Systems Lab



1. Motivación
2. Trabajos previos
3. Descomposición en SPMs
4. Compilación incremental basada en DSPM
 - (a) Descripción general
 - (b) Detalle de algunos casos y algoritmos
5. Conclusiones

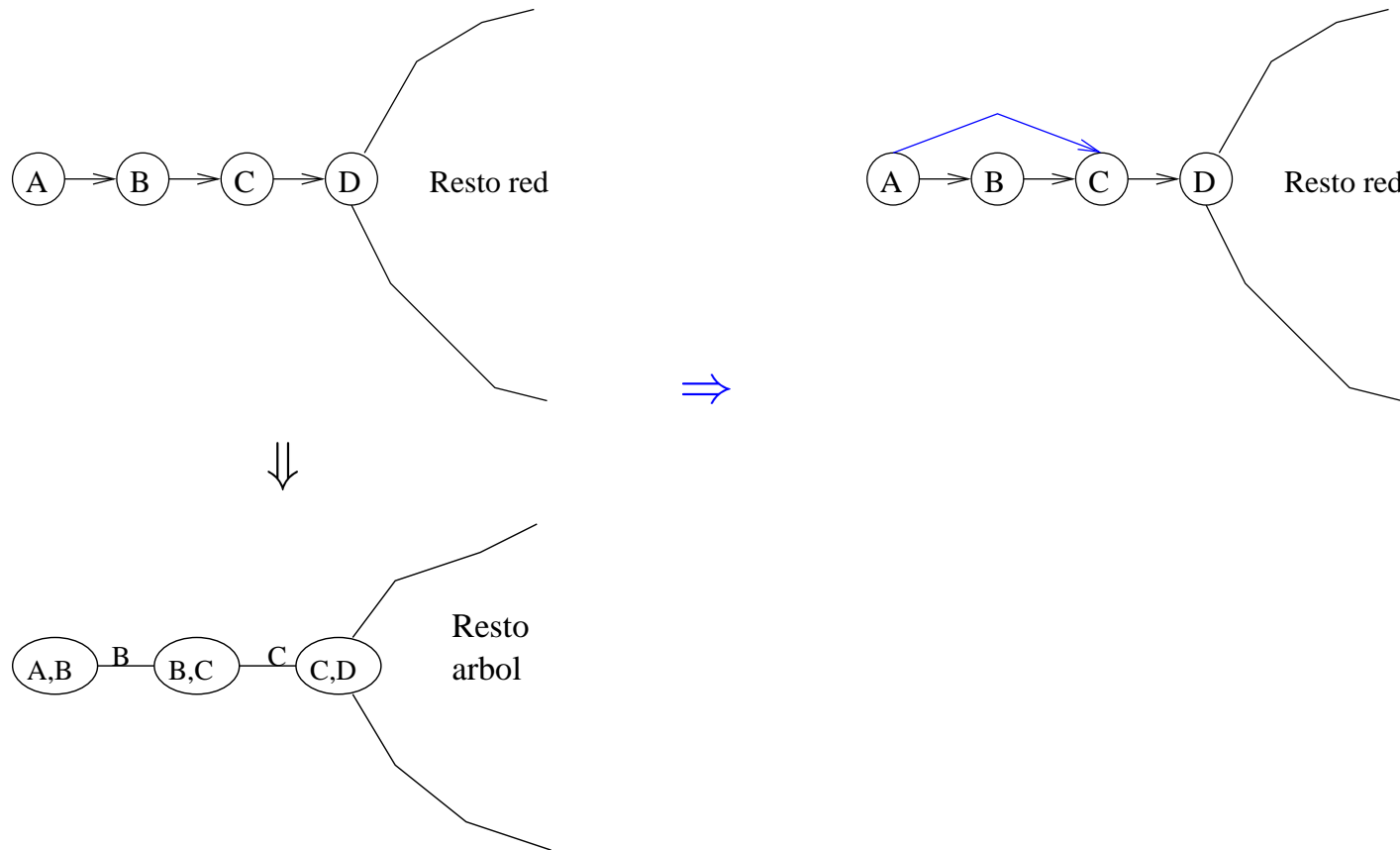
Motivación

- Cada vez que se pasa de la ventana de edición a la ventana de ejecución un proceso completo de triangulación/compilación es realizado



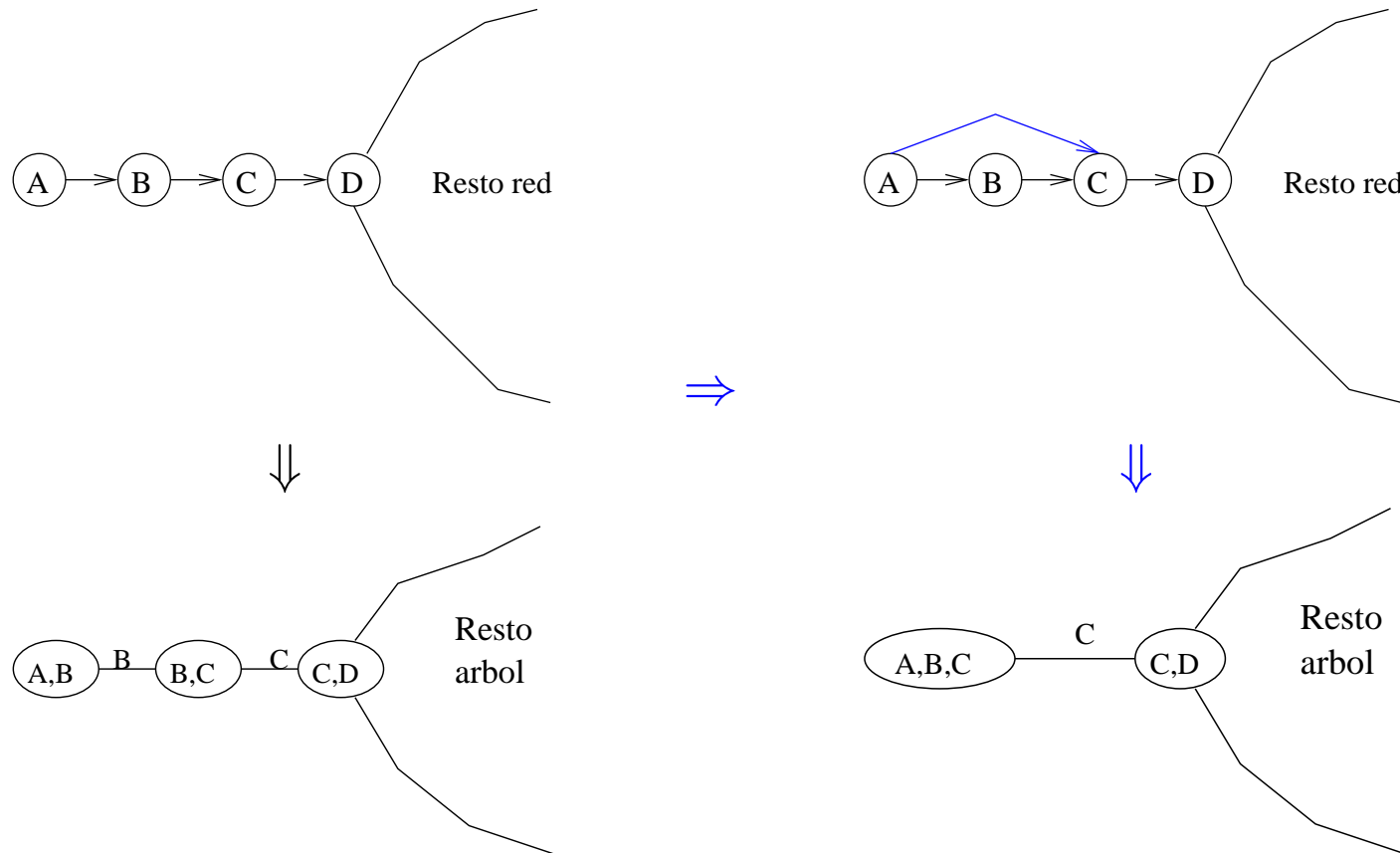
Motivación

- Cada vez que se pasa de la ventana de edición a la ventana de ejecución un proceso completo de triangulación/compilación es realizado



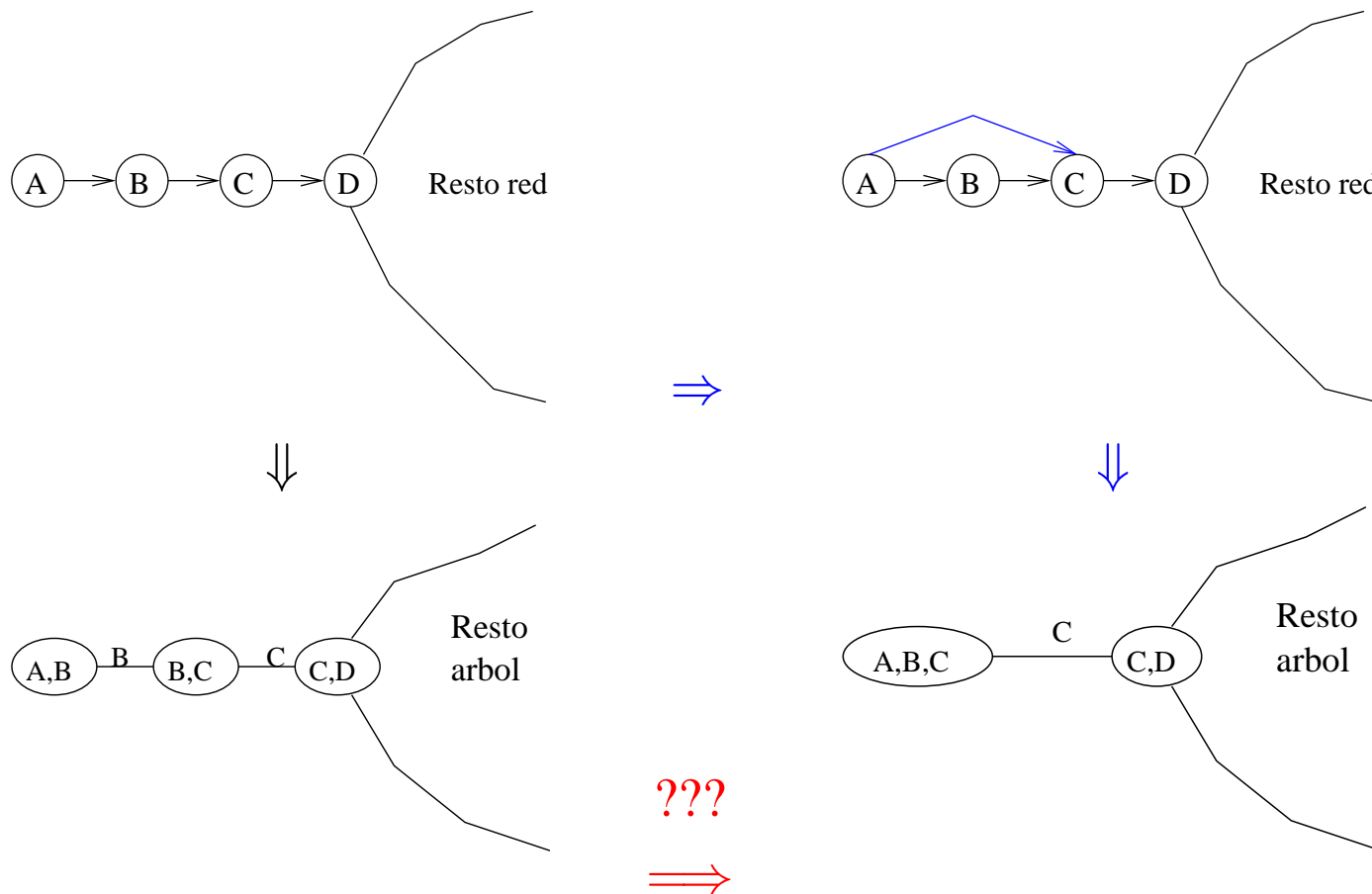
Motivación

- Cada vez que se pasa de la ventana de edición a la ventana de ejecución un **proceso completo de triangulación/compilación** es realizado
- Triangular es un proceso costoso. ¿Podría evitarse **al menos parcialmente**?



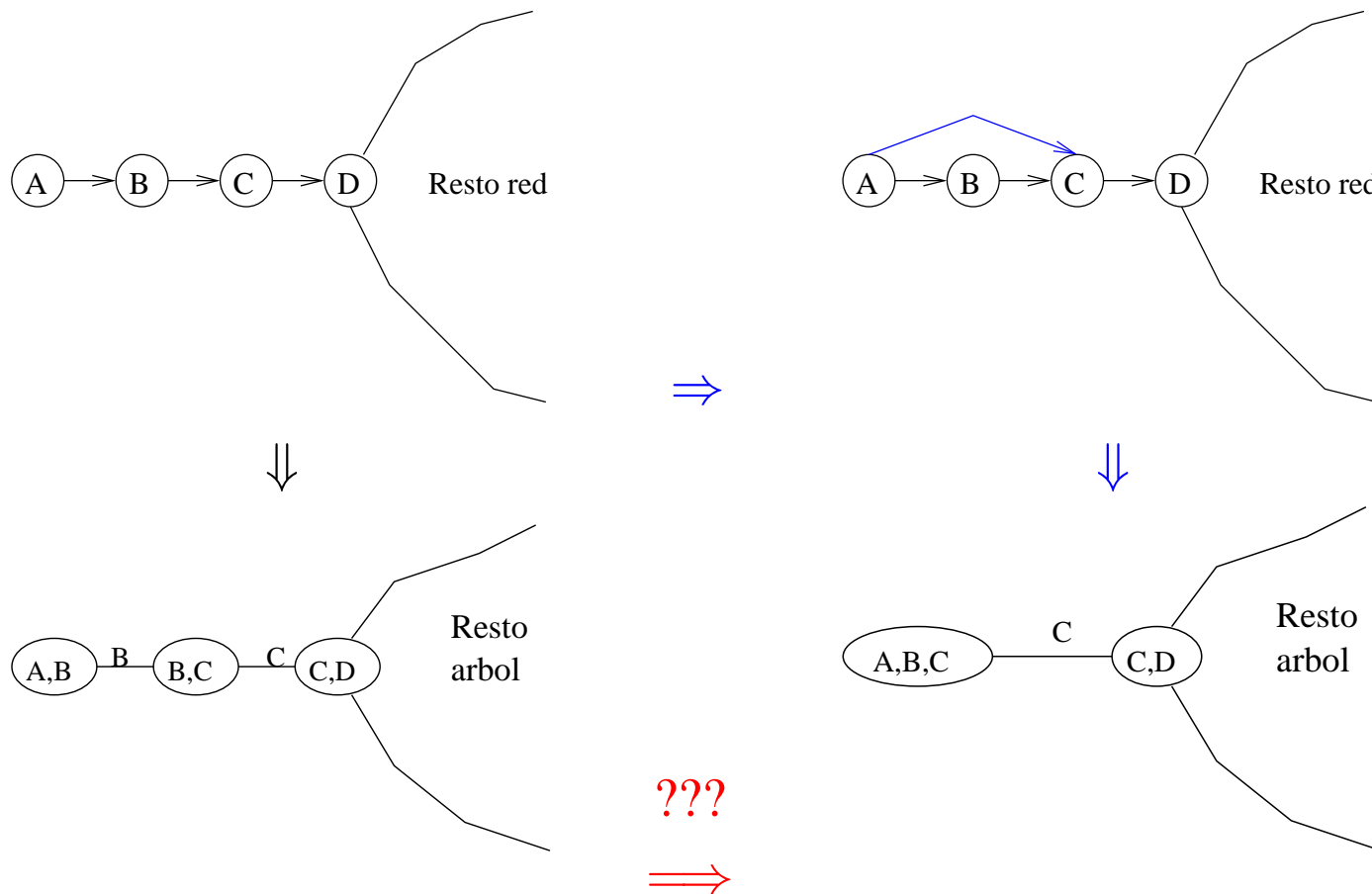
Motivación

- Cada vez que se pasa de la ventana de edición a la ventana de ejecución un **proceso completo de triangulación/compilación** es realizado
- Triangular es un proceso costoso. ¿Podría evitarse **al menos parcialmente**?



Motivación

- Cada vez que se pasa de la ventana de edición a la ventana de ejecución un **proceso completo de triangulación/compilación** es realizado
- Triangular es un proceso costoso. ¿Podría evitarse **al menos parcialmente**?
- Ventajas de la compilación incremental: Tiempo, estabilidad, ...



Trabajos previos

- A. Darwiche. *Dynamic Jointrees*. UAI'98.
- D. Draper. *Clustering Without (Thinking About) Triangulation*. UAI'95.

Trabajos previos

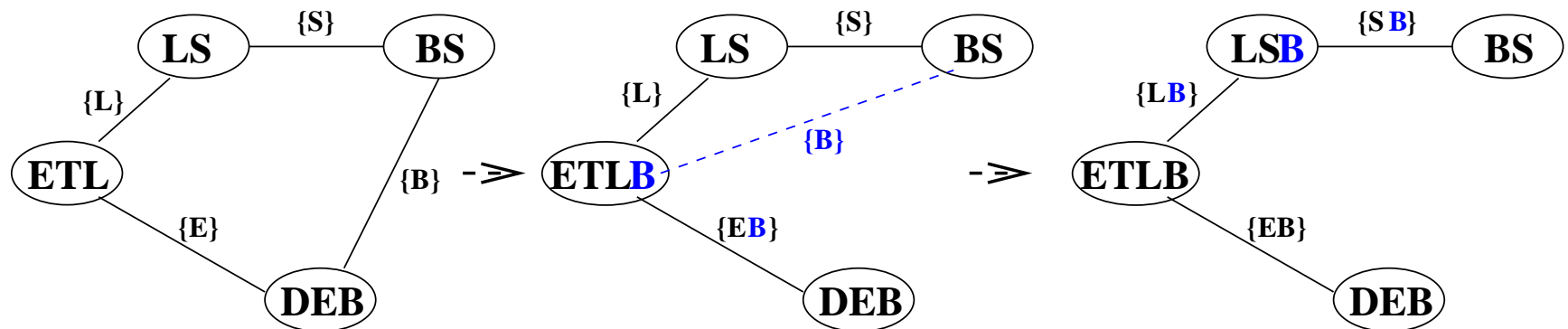
- A. Darwiche. *Dynamic Jointrees*. UAI'98.
 - ▶ Modificación dinámica de árboles de cliques para obtener estructuras eficientes con respecto a un problema (*query*) concreto.
- D. Draper. *Clustering Without (Thinking About) Triangulation*. UAI'95.

Trabajos previos

- A. Darwiche. *Dynamic Jointrees*. UAI'98.
 - ▶ Modificación dinámica de árboles de cliques para obtener estructuras eficientes con respecto a un problema (*query*) concreto.
- D. Draper. *Clustering Without (Thinking About) Triangulation*. UAI'95.
 - ▶ Principal objetivo del artículo: construir árboles de cliques sin triangular. Parte de un grafo de familias y mediante un conjunto de operaciones de transformación obtiene un árbol de cliques.

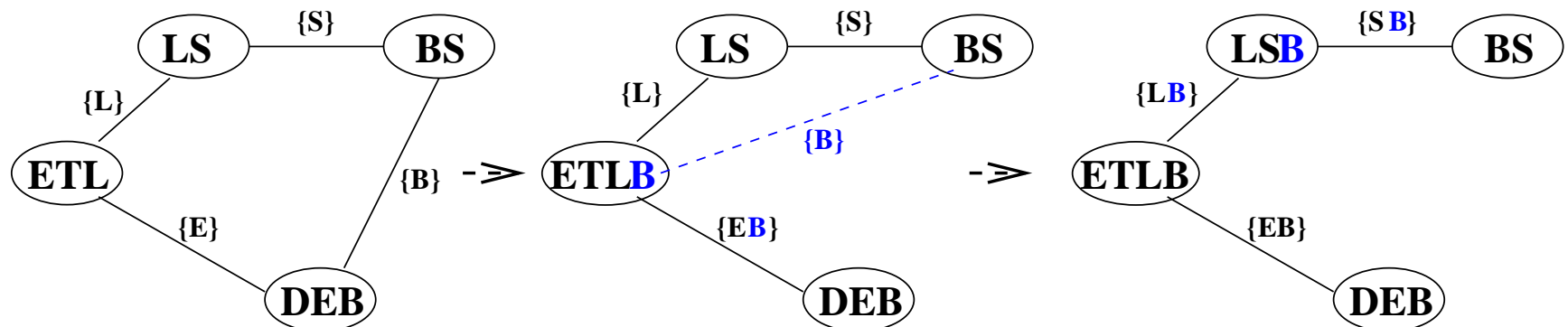
Trabajos previos

- A. Darwiche. *Dynamic Jointrees*. UAI'98.
 - ▶ Modificación dinámica de árboles de cliques para obtener estructuras eficientes con respecto a un problema (*query*) concreto.
- D. Draper. *Clustering Without (Thinking About) Triangulation*. UAI'95.
 - ▶ Principal objetivo del artículo: construir árboles de cliques sin triangular. Parte de un grafo de familias y mediante un conjunto de operaciones de transformación obtiene un árbol de cliques.



Trabajos previos

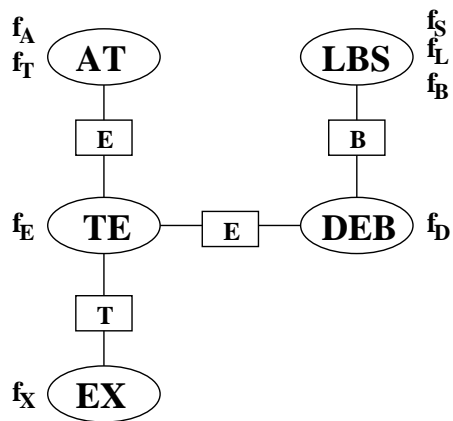
- A. Darwiche. *Dynamic Jointrees*. UAI'98.
 - ▶ Modificación dinámica de árboles de cliques para obtener estructuras eficientes con respecto a un problema (*query*) concreto.
- D. Draper. *Clustering Without (Thinking About) Triangulation*. UAI'95.
 - ▶ Principal objetivo del artículo: construir árboles de cliques sin triangular. Parte de un grafo de familias y mediante un conjunto de operaciones de transformación obtiene un árbol de cliques.



- ▶ Algunas de las transformaciones propuestas son aplicadas para realizar compilación incremental.

Compilación incremental: Enfoque de Draper

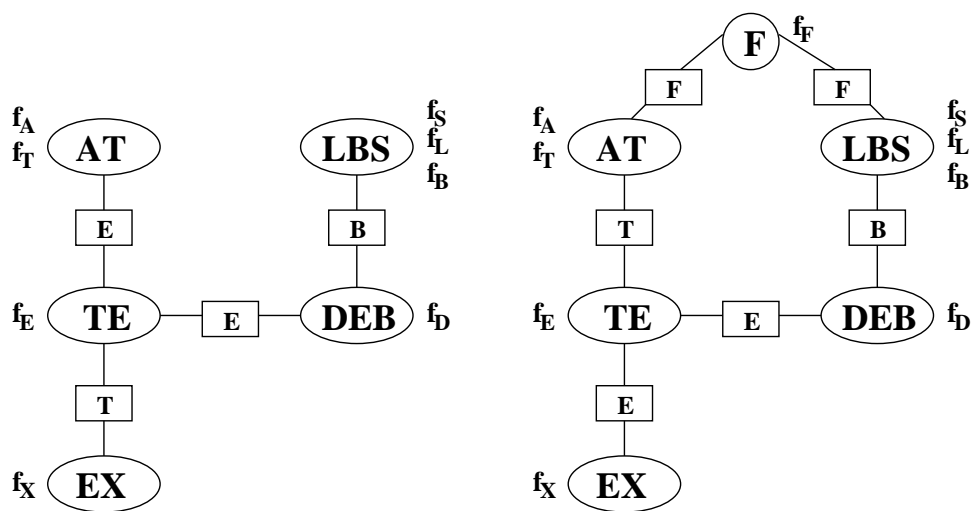
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

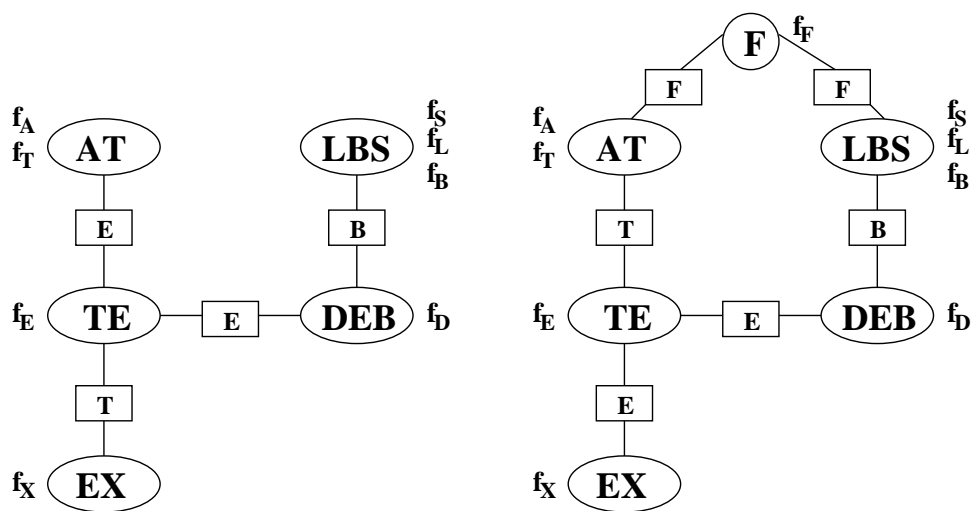
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
⇒ posiblemente obtener un **grafo** de grupos



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

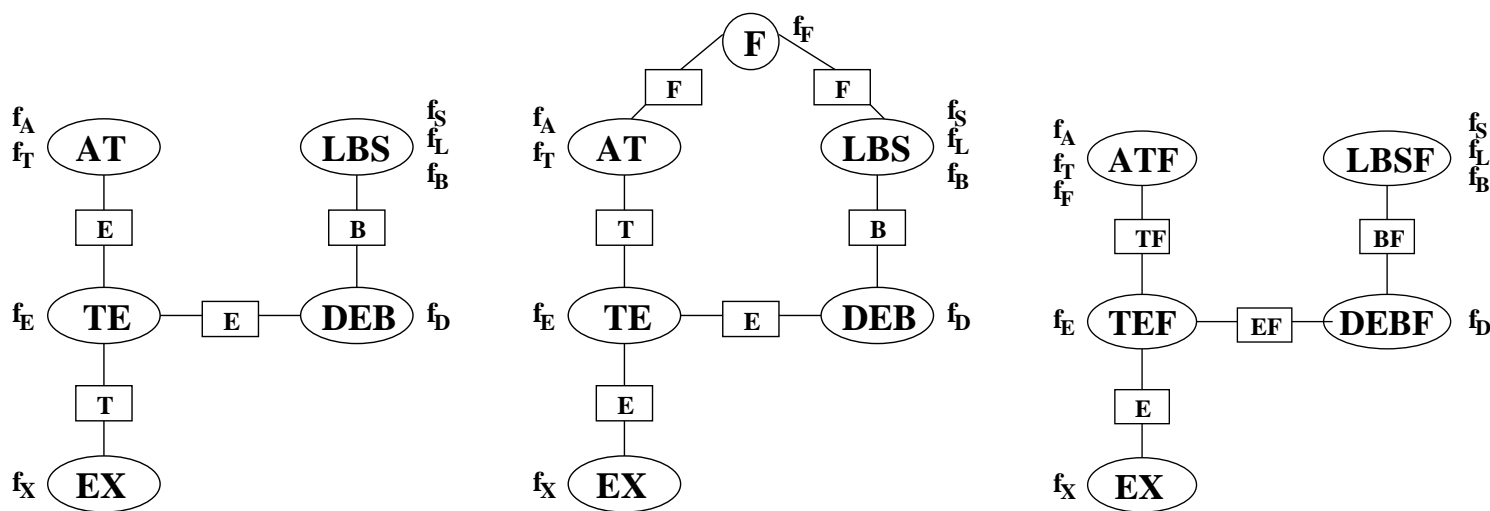
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
⇒ posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

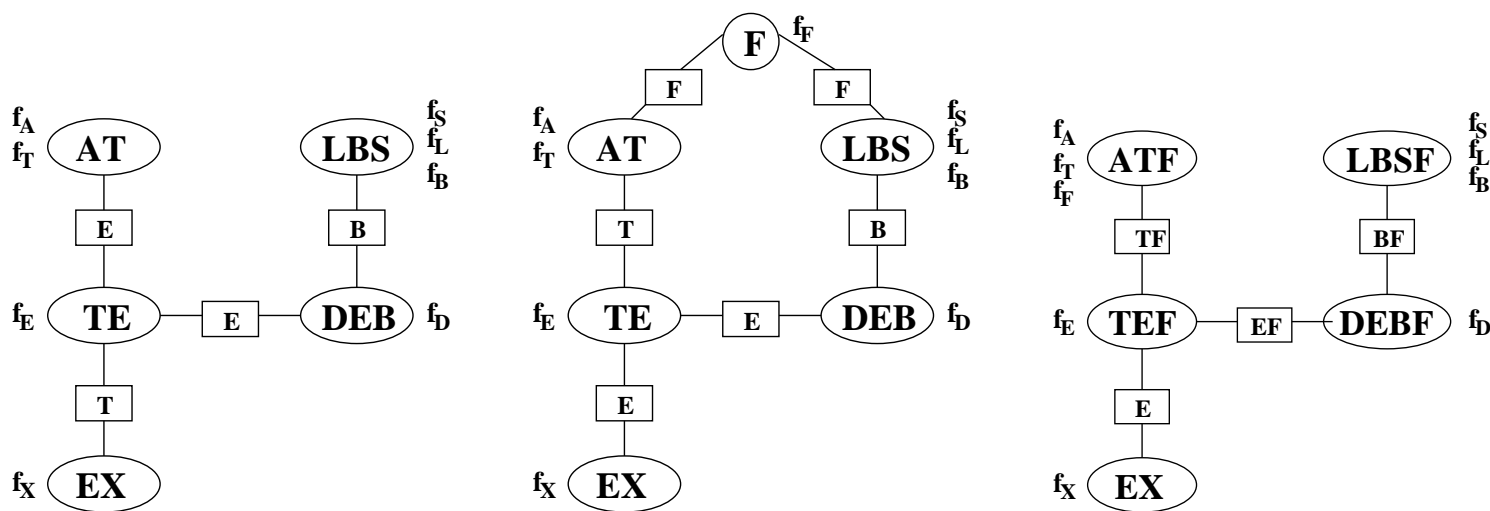
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
 \Rightarrow posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

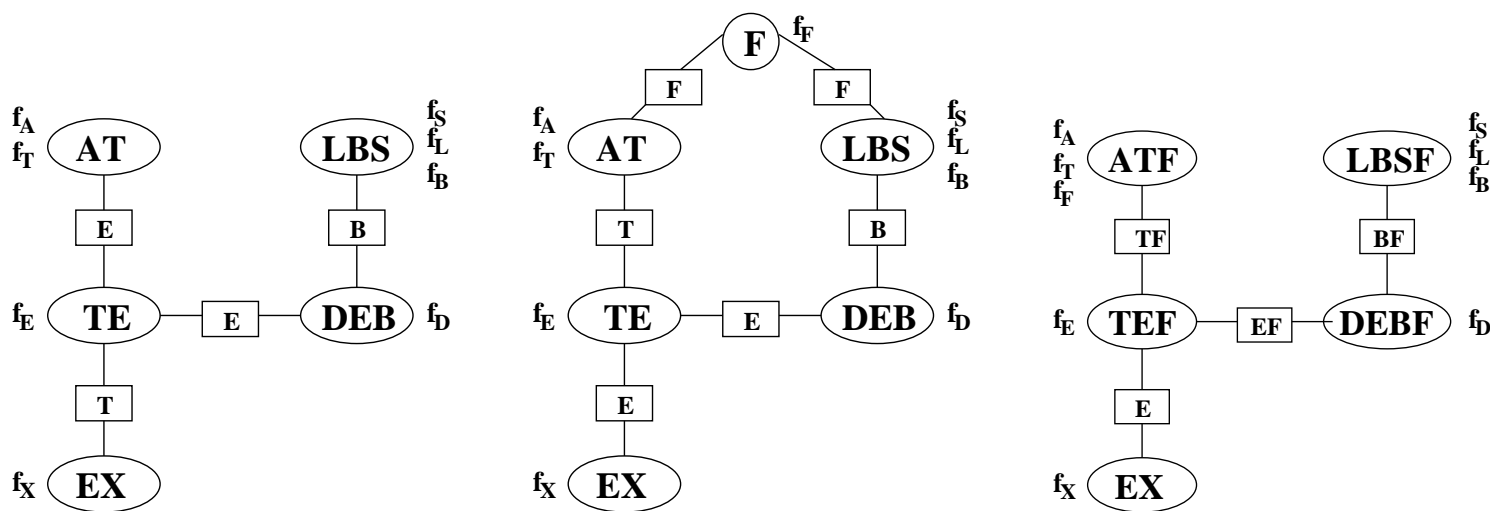
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
 - ⇒ posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol
- Comentarios:
 - ligero *tufillo* a triangulación en las transformaciones introducidas para romper ciclos.



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

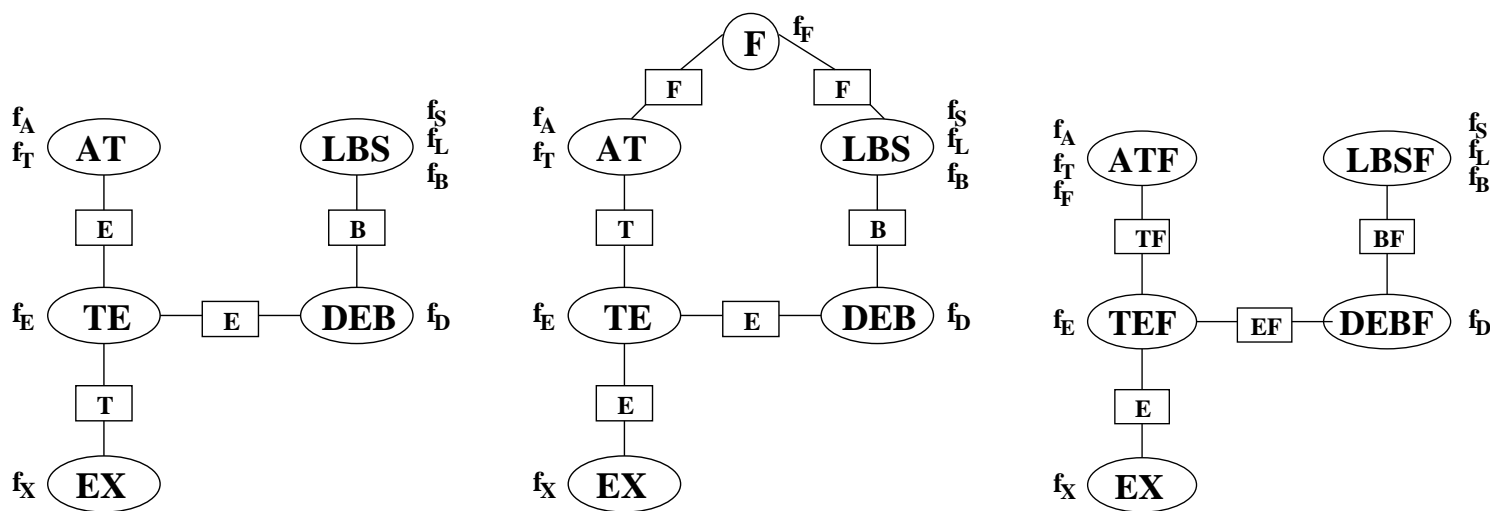
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
 - ⇒ posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol
- Comentarios:
 - ligero *tufillo* a triangulación en las transformaciones introducidas para romper ciclos.
 - muchos grados de libertad => heurísticas



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

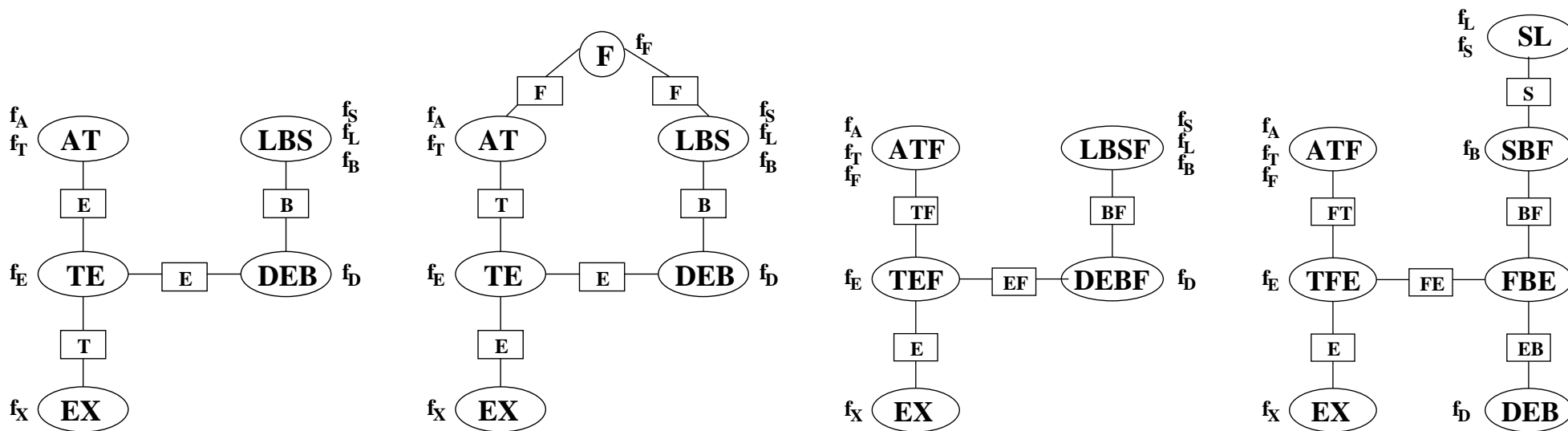
- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
⇒ posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol
- Comentarios:
 - ligero *tufillo* a triangulación en las transformaciones introducidas para romper ciclos.
 - muchos grados de libertad => heurísticas
 - algunos pasos no son triviales y los resultados no son muy buenos



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

Compilación incremental: Enfoque de Draper

- Imputar los cambios estructurales realizados sobre la red, al **árbol** de grupos.
 - ⇒ posiblemente obtener un **grafo** de grupos
- Aplicar las transformaciones introducidas para volver a obtener un árbol
- Comentarios:
 - ligero *tufillo* a triangulación en las transformaciones introducidas para romper ciclos.
 - muchos grados de libertad => heurísticas
 - algunos pasos no son triviales y los resultados no son muy buenos



Añadiendo $T \leftarrow F \rightarrow S$ a Asia menos $L \rightarrow E$.

- Realizar compilación incremental basándonos en técnicas de triangulación

Objetivos

- Realizar compilación incremental basándonos en técnicas de **triangulación**
- Considerar sólo cambios estructurales en la red

Objetivos

- Realizar compilación incremental basándonos en técnicas de **triangulación**
- Considerar sólo cambios estructurales en la red
- Si bien el usuario puede pasar una lista de modificaciones, el algoritmo itera sobre modificaciones básicas

Objetivos

- Realizar compilación incremental basándonos en técnicas de **triangulación**
- Considerar sólo cambios estructurales en la red
- Si bien el usuario puede pasar una lista de modificaciones, el algoritmo itera sobre modificaciones básicas
 - ▶ Añadir nodo/enlace

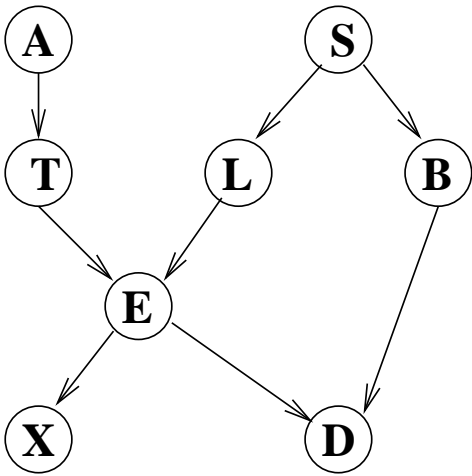
- Realizar compilación incremental basándonos en técnicas de **triangulación**
- Considerar sólo cambios estructurales en la red
- Si bien el usuario puede pasar una lista de modificaciones, el algoritmo itera sobre modificaciones básicas
 - ▶ Añadir nodo/enlace
 - ▶ Borrar nodo/enlace

- Realizar compilación incremental basándonos en técnicas de **triangulación**
- Considerar sólo cambios estructurales en la red
- Si bien el usuario puede pasar una lista de modificaciones, el algoritmo itera sobre modificaciones básicas
 - ▶ Añadir nodo/enlace
 - ▶ Borrar nodo/enlace
- Usar la **descomposición en subgrafos primos maximales (DSPM)** como estructura intermedia.

Descomposición en subgrafos primos maximales

- Obtención de la DSPM:

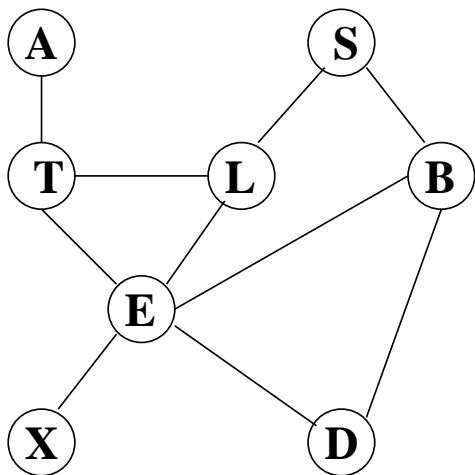
G



Descomposición en subgrafos primos maximales

- Obtención de la DSPM:

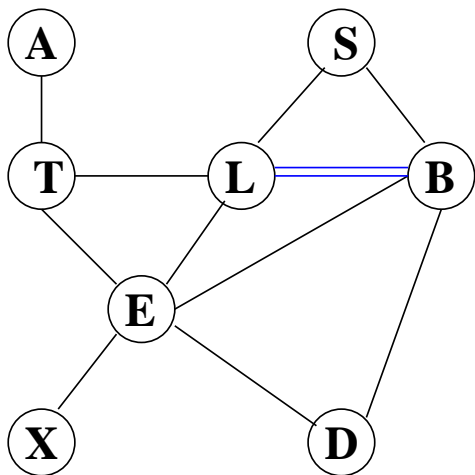
$$G \longrightarrow G^M$$



Descomposición en subgrafos primos maximales

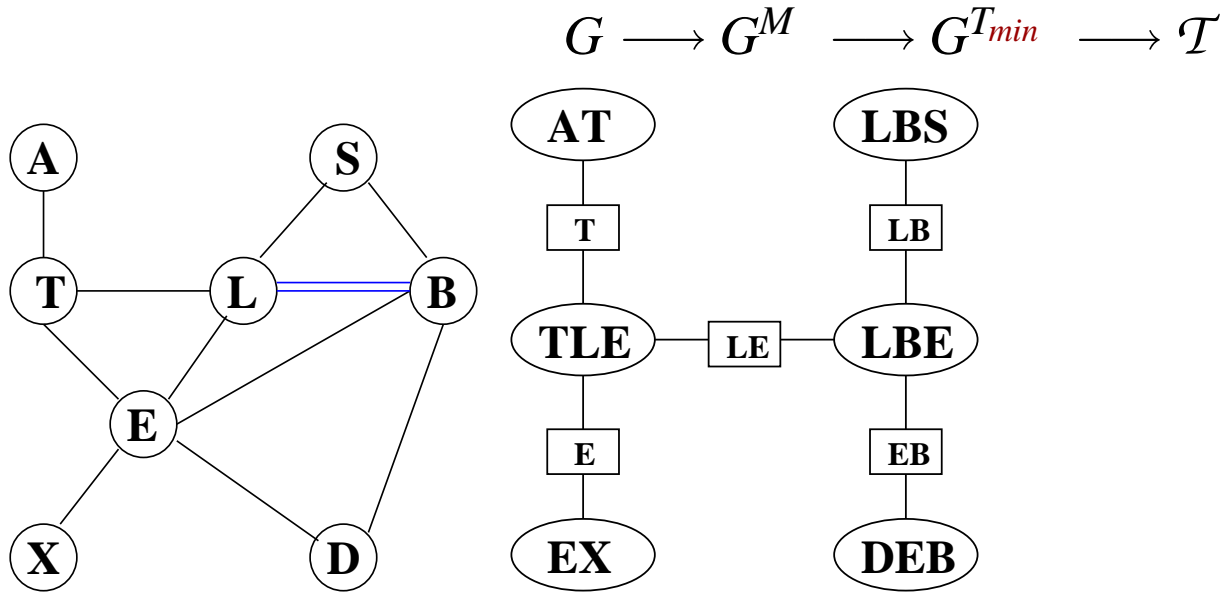
- Obtención de la DSPM:

$$G \longrightarrow G^M \longrightarrow G^{T_{min}}$$



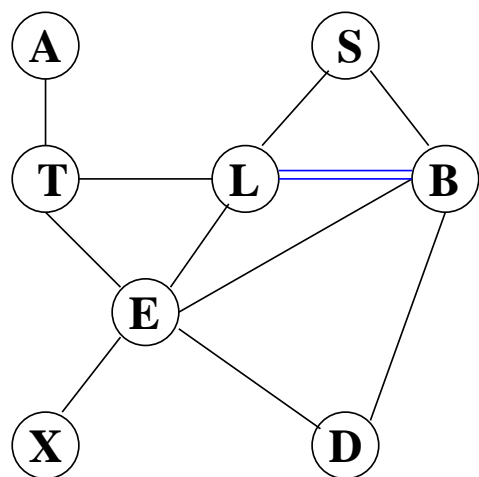
Descomposición en subgrafos primos maximales

■ Obtención de la DSPM:

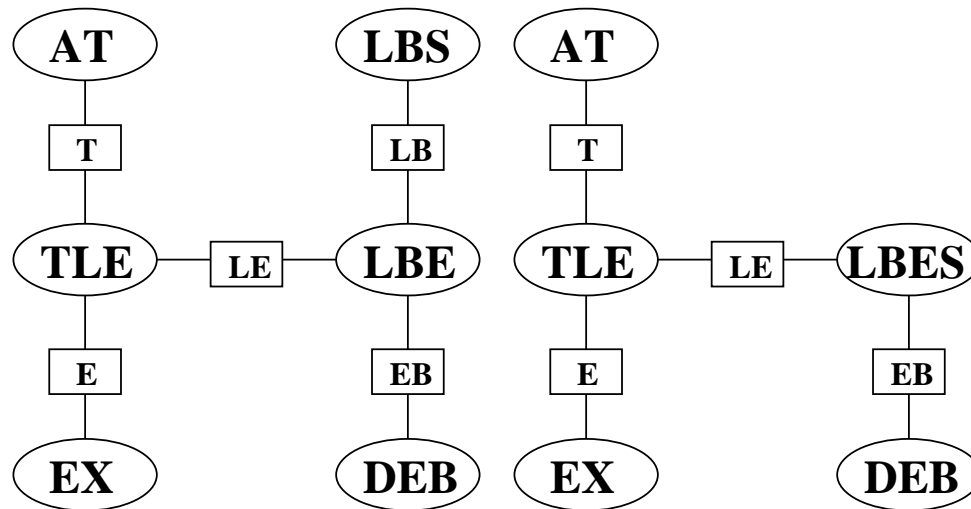


Descomposición en subgrafos primos maximales

■ Obtención de la DSPM:

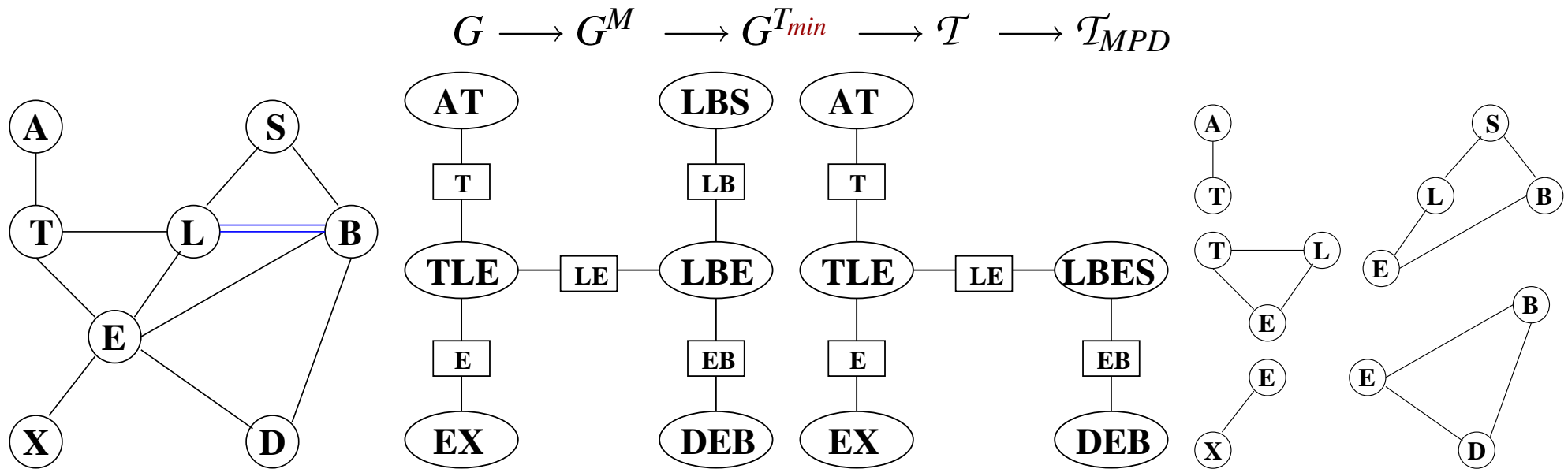


$$G \longrightarrow G^M \longrightarrow G^{T_{min}} \longrightarrow \mathcal{T} \longrightarrow \mathcal{T}_{MPD}$$



Descomposición en subgrafos primos maximales

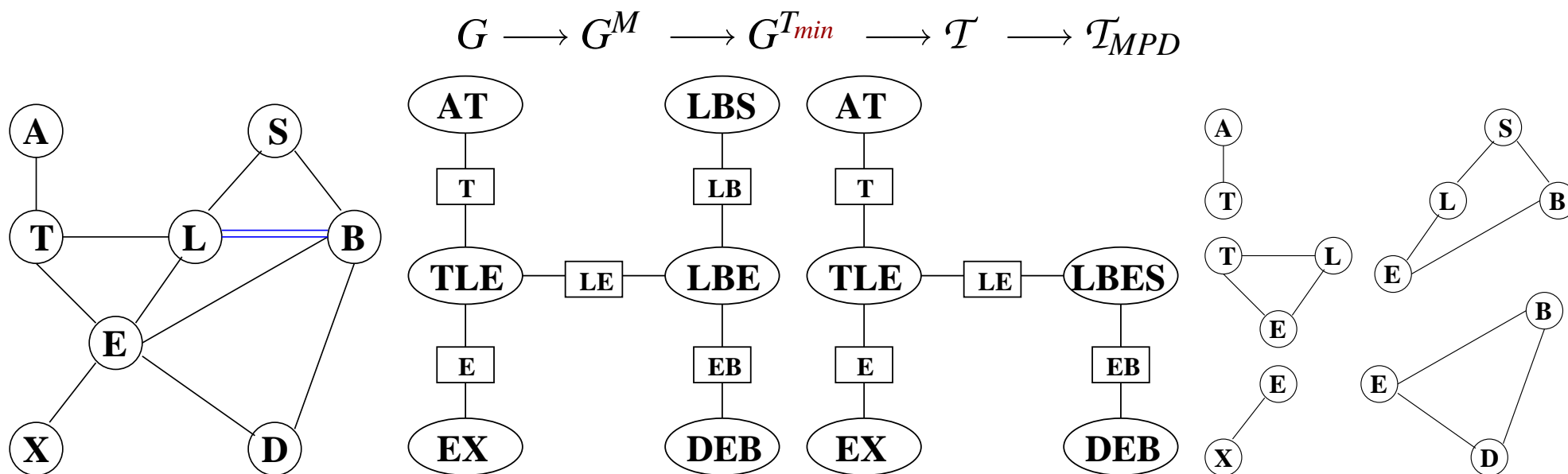
- Obtención de la DSPM:



- La principal ventaja de la DSPM es que **cada subgrafo puede ser triangulado independientemente**

Descomposición en subgrafos primos maximales

- Obtención de la DSPM:



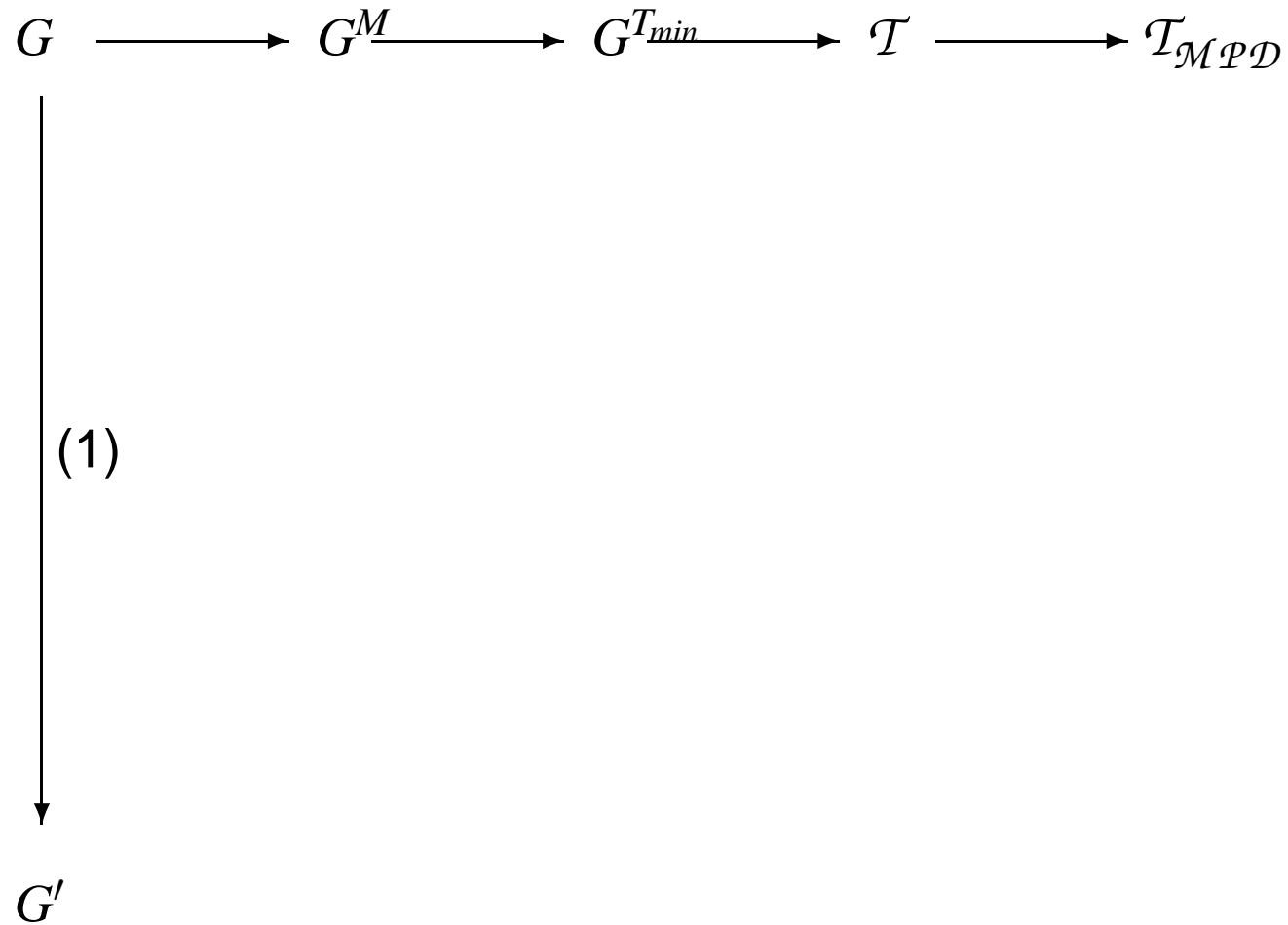
- La principal ventaja de la DSPM es que **cada subgrafo puede ser triangulado independientemente**
- Nuestro **objetivo** es identificar el número mínimo de SMPs que hay que retriangular, hacerlo y luego fusionar las estructuras gráficas.

Compilación incremental basada en DSPM

$$G \longrightarrow G^M \longrightarrow G^{T_{min}} \longrightarrow \mathcal{T} \longrightarrow \mathcal{T}_{MPD}$$

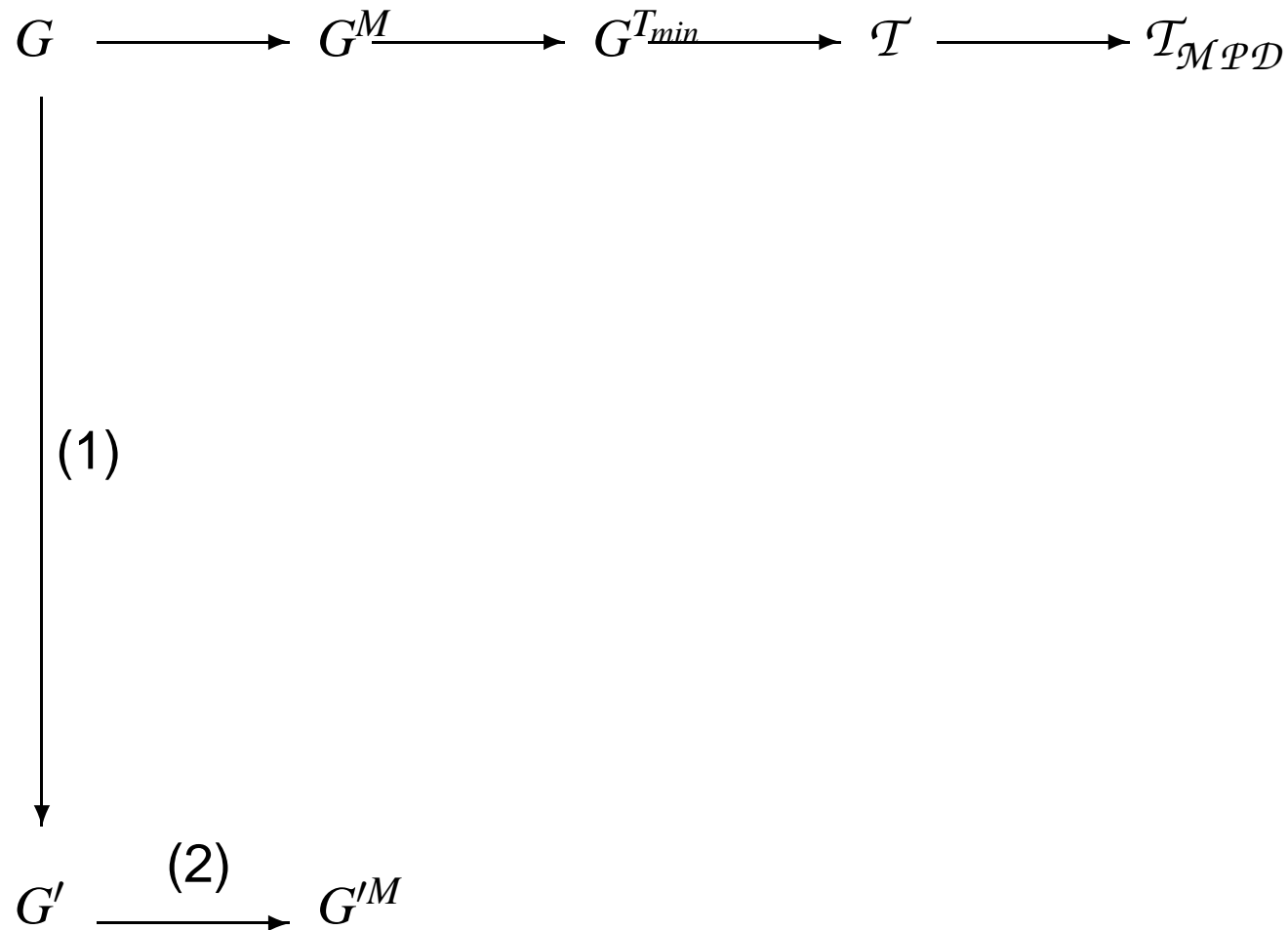
- Situación de partida

Compilación incremental basada en DSPM



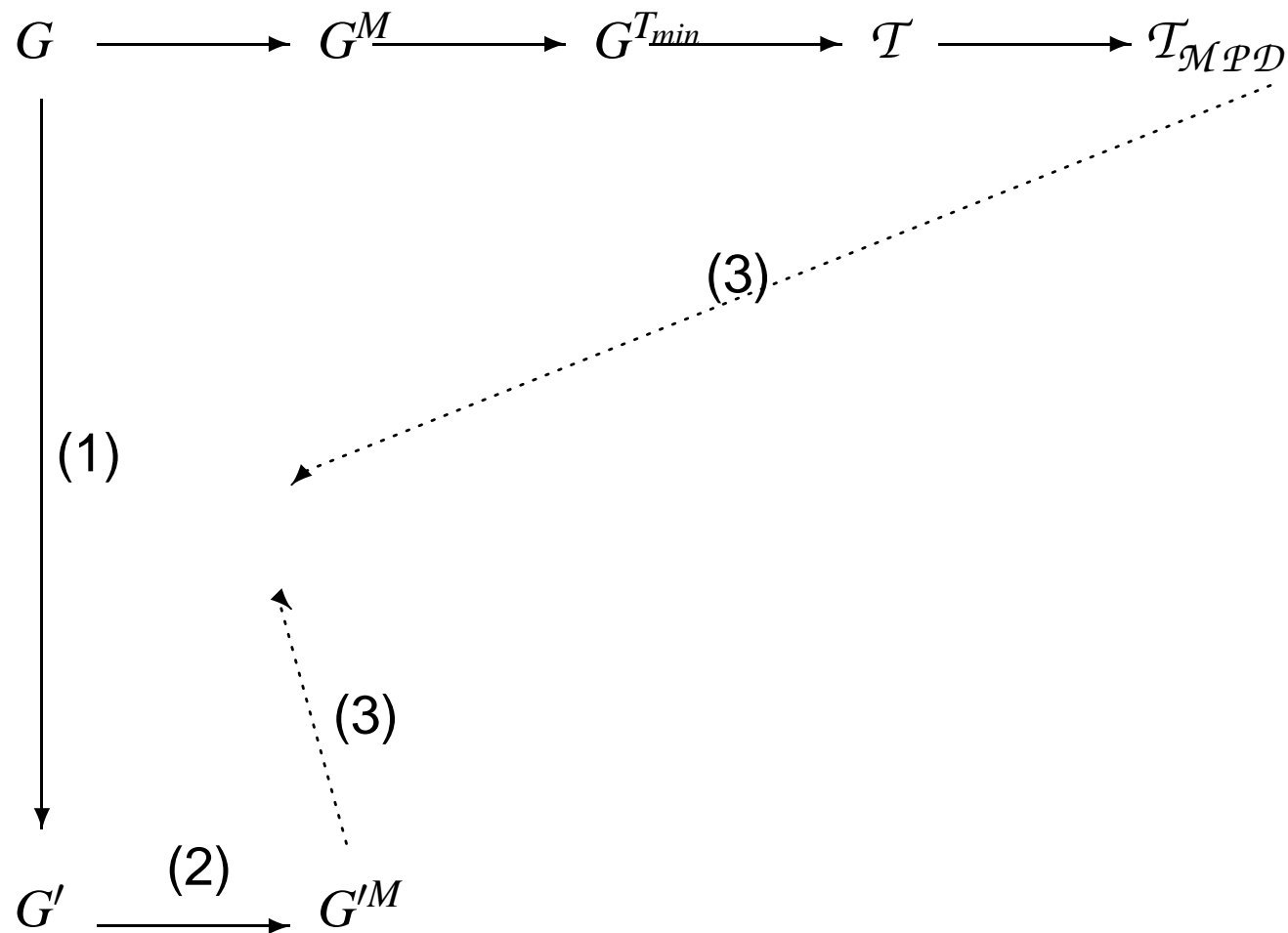
- Modificación(es) sobre la red

Compilación incremental basada en DSPM



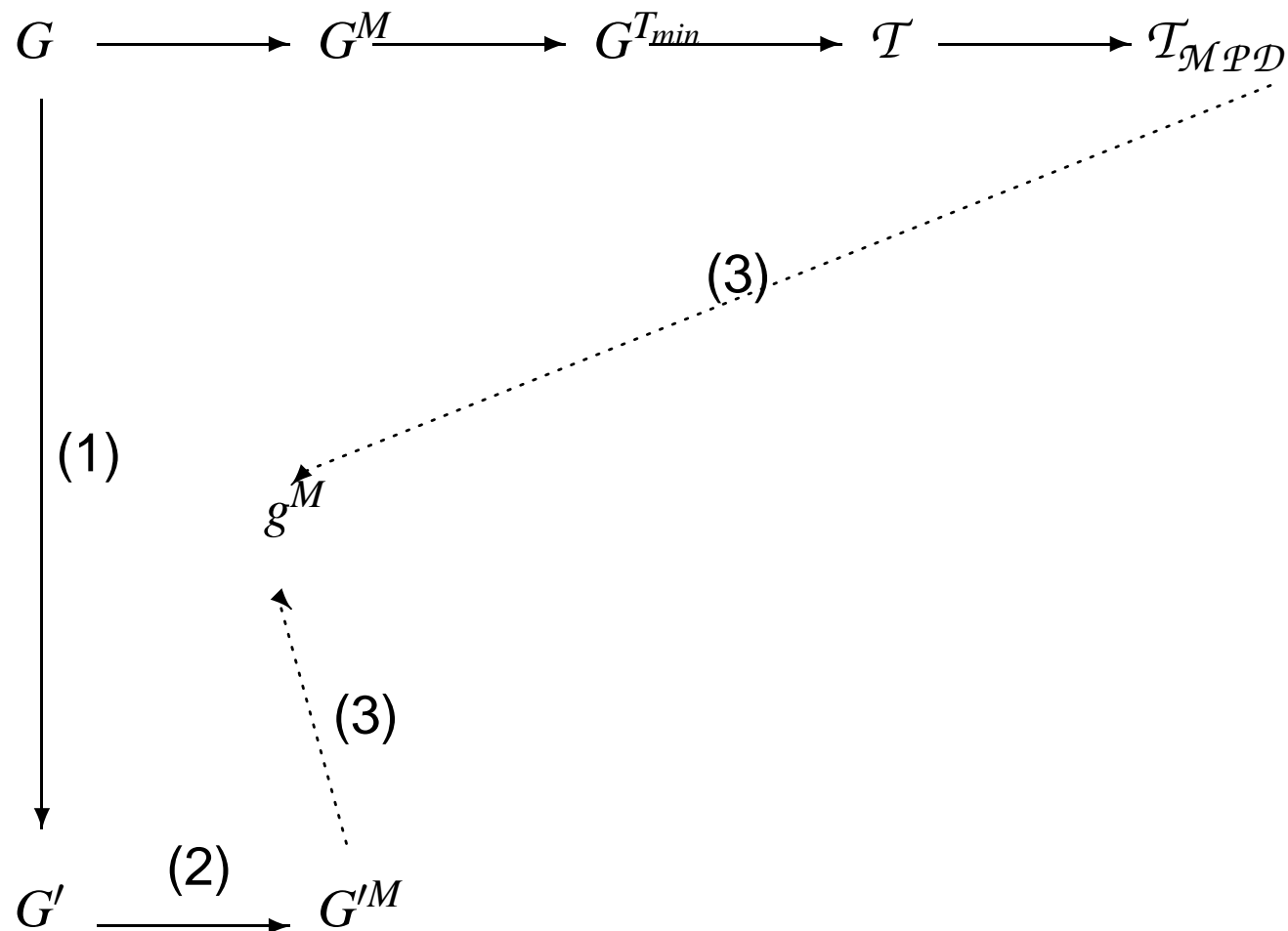
- Imputamos las modificaciones al grafo moral

Compilación incremental basada en DSPM



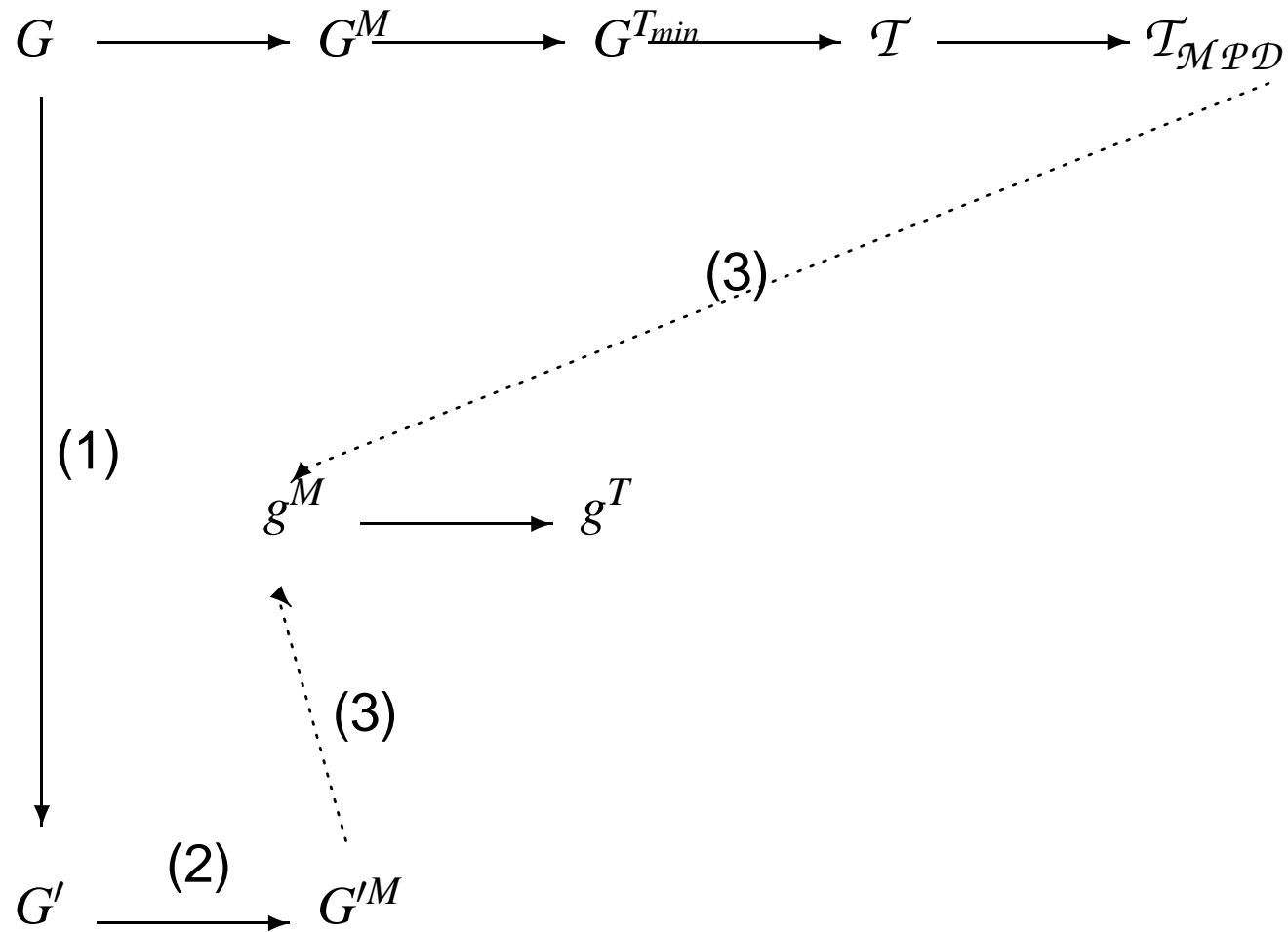
- Usando el conjunto de aristas modificadas en el grafo moral, se marca el conjunto mínimo de SPMs en T_{MPD}

Compilación incremental basada en DSPM



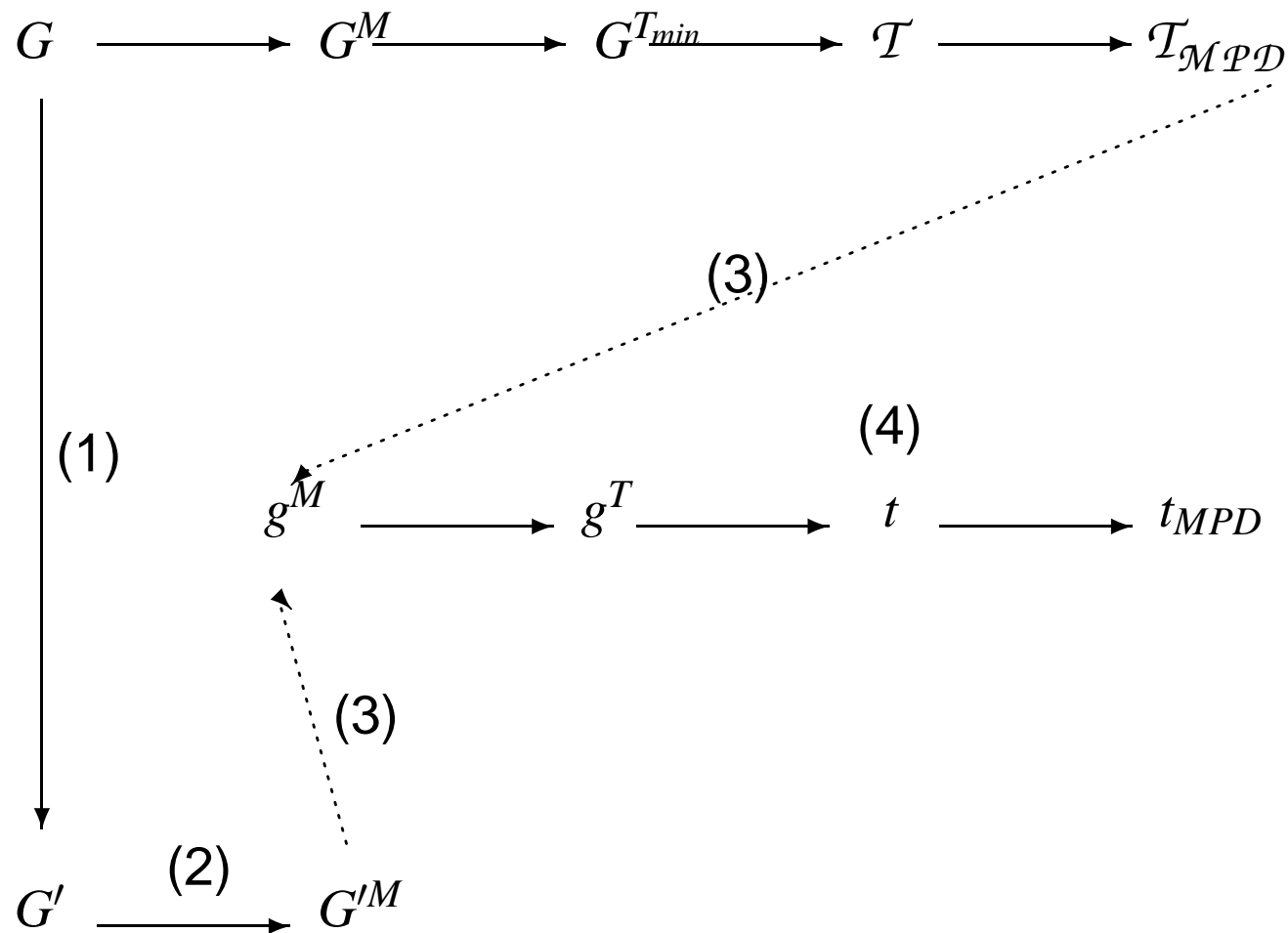
- g^M es la proyección del conjunto de variables contenido en los SPMs marcados sobre G'^M

Compilación incremental basada en DSPM



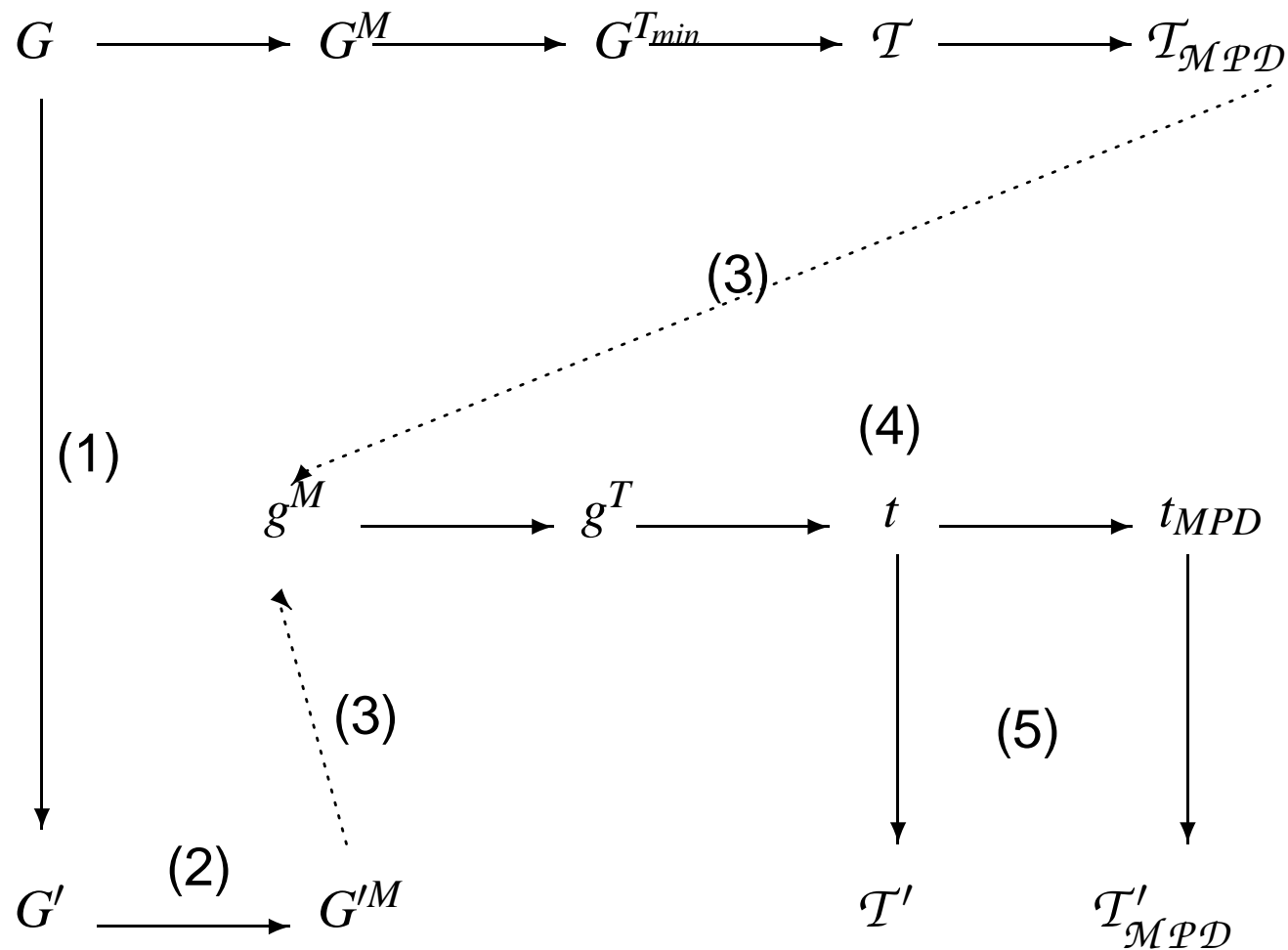
- Triangular g^M . Notar que g^T debe ser una triangulación minimal.

Compilación incremental basada en DSPM



- Obtener un árbol de grupos y un árbol de SPMs para g^T con respecto a g^M

Compilación incremental basada en DSPM



- Conectar las nuevas estructuras con las viejas y borrar todos los SPMs marcados (y en \mathcal{T} los cliques en ellos incluidos)

Algoritmo: (DSPM)-Compilación incremental

Algorithm 1 IncrementalCompilation (Modification list *ModList*)

1. For each modification *mod* in *ModList* do

Algoritmo: (DSPM)-Compilación incremental

Algorithm 1 IncrementalCompilation (Modification list $ModList$)

1. For each modification mod in $ModList$ do
 - (a) $L \leftarrow \text{ModifyMoralGraph}(mod)$

Algoritmo: (DSPM)-Compilación incremental

Algorithm 1 IncrementalCompilation (Modification list $ModList$)

1. For each modification mod in $ModList$ do
 - (a) $L \leftarrow \text{ModifyMoralGraph}(mod)$
 - (b) Case mod of
 - i. Add node X : $\text{AddNode}(X)$
 - ii. Delete node X : $\text{RemoveNode}(X, M_X, nil)$
 - iii. Delete link $X \rightarrow Y$: $\text{MarkAffectedMPSsByRemoveLink}(M_Y, nil, L)$
 - iv. Add link $X \rightarrow Y$: $\text{MarkAffectedMPSsByAddingLink}(L)$

Algoritmo: (DSPM)-Compilación incremental

Algorithm 1 IncrementalCompilation (Modification list $ModList$)

1. For each modification mod in $ModList$ do
 - (a) $L \leftarrow \text{ModifyMoralGraph}(mod)$
 - (b) Case mod of
 - i. Add node X : $\text{AddNode}(X)$
 - ii. Delete node X : $\text{RemoveNode}(X, M_X, nil)$
 - iii. Delete link $X \rightarrow Y$: $\text{MarkAffectedMPSsByRemoveLink}(M_Y, nil, L)$
 - iv. Add link $X \rightarrow Y$: $\text{MarkAffectedMPSsByAddingLink}(L)$
2. For each connected marked subtree, T_{MPD} , of \mathcal{T}_{MPD} do
 - (a) Mark all cliques in the subtree T of \mathcal{T} corresponding to T_{MPD}
 - (b) Let C be any cluster of T and let M be any cluster of T_{MPD}
 - (c) $V \leftarrow \{\text{all variables included in } T_{MPS}\}$
 - (d) $g^M \leftarrow G^M(V)$
 - (e) $t \leftarrow \text{ConstructJoinTree}(g^M)$
 - (f) $t_{MPD} \leftarrow \text{AggregateCliques}(t)$
 - (g) $\mathcal{T} \leftarrow \text{connect}(t, C, nil)$
 - (h) $\mathcal{T}_{MPD} \leftarrow \text{connect}(t_{MPD}, M, nil)$
 - (i) Delete T and T_{MPD}

Algoritmo: Modificar Grafo Moral

Algorithm 2 ModifyMoralGraph(Modification mod)

1. $L \leftarrow \emptyset$ // a link list
2. Case mod of
 - (a) Add node X : add a new (isolated) node X to G^M
 - (b) Delete node X :
 - i. for each link l containing X do
 - $L \leftarrow L \cup \text{ModifyMoralGraph}(\text{remove } l)$
 - ii. remove X from G^M
 - (c) Add link $X \rightarrow Y$: add $X \rightarrow Y$ to L together with all new links needed to make $Y \cup \text{parents}(Y)$ a complete sub-graph
 - (d) Delete link $X \rightarrow Y$:
 - i. if $(\text{children}(X) \cap \text{children}(Y) = \emptyset)$ then
 - delete (X, Y) from G^M
 - add (X, Y) to L
 - ii. for all $Z_i \in \text{parents}(Y) \setminus \{X\}$ do
 - A. if $(\text{children}(Z_i) \cap \text{children}(X) = \{Y\})$ and $Z_i \rightarrow X$ in G and $X \rightarrow Z_i$ not in G then
 - delete (X, Z_i) from G^M and add (X, Z_i) to L
3. Return L

Ejemplo: borrando un enlace

- Para borrar $X \rightarrow Y$, localizamos el SPM M_Y que contiene a f_Y .

Algorithm 3 MarkAffectedMPSsByRemoveLink(MPS M_Y , MPS M_Z , LinkList L)

1. Mark M_Y

Ejemplo: borrando un enlace

- Para borrar $X \rightarrow Y$, localizamos el SPM M_Y que contiene a f_Y .
- Vemos si alguno de los separadores de M_Y se ve afectado por la lista de enlaces eliminados (L).

Algorithm 3 MarkAffectedMPSsByRemoveLink(MPS M_Y , MPS M_Z , LinkList L)

1. Mark M_Y
2. For all neighbors $M_K \neq M_Z$ of M_Y do
 - (a) $S_{YK} \leftarrow$ separator between M_Y and M_K

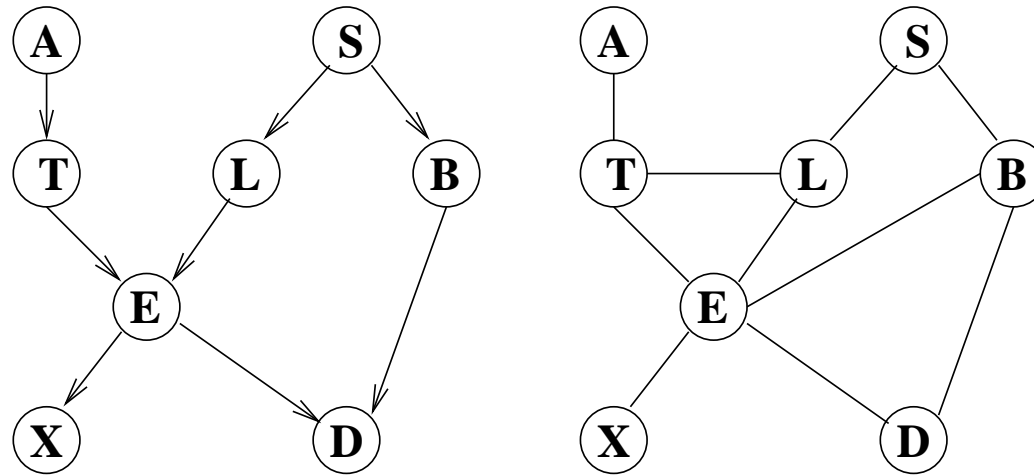
Ejemplo: borrando un enlace

- Para borrar $X \rightarrow Y$, localizamos el SPM M_Y que contiene a f_Y .
- Vemos si alguno de los separadores de M_Y se ve afectado por la lista de enlaces eliminados (L).
- En caso afirmativo se continúa la búsqueda recursivamente

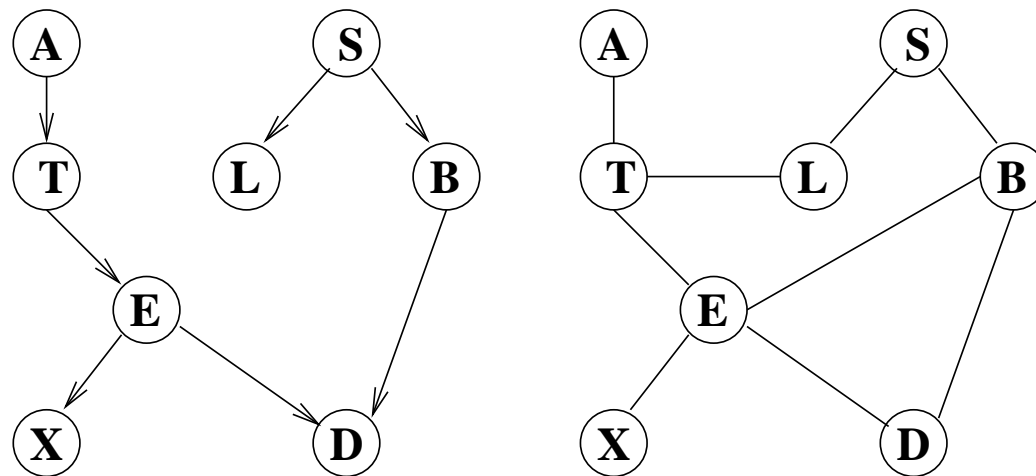
Algorithm 3 MarkAffectedMPSsByRemoveLink(MPS M_Y , MPS M_Z , LinkList L)

1. Mark M_Y
2. For all neighbors $M_K \neq M_Z$ of M_Y do
 - (a) $S_{YK} \leftarrow$ separator between M_Y and M_K
 - (b) if $L \cap \text{links}(S_{YK}) \neq \emptyset$ then
MarkAffectedMPSsByRemoveLink(M_K, M_Y, L)

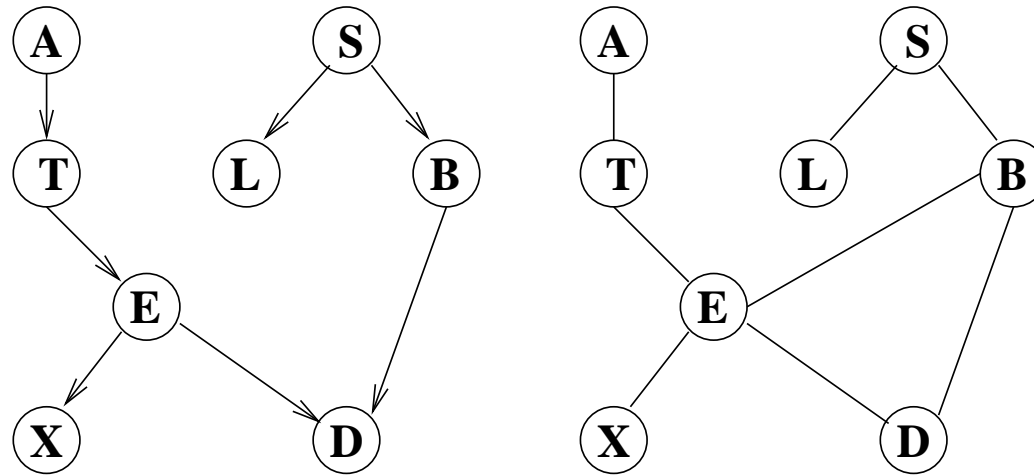
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia



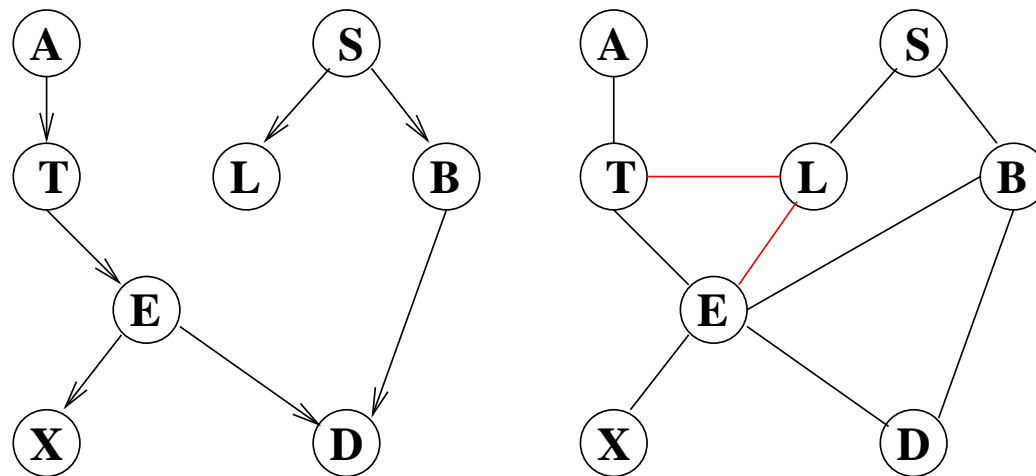
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia



Ejemplo: borrando el enlace $L \rightarrow E$ en Asia



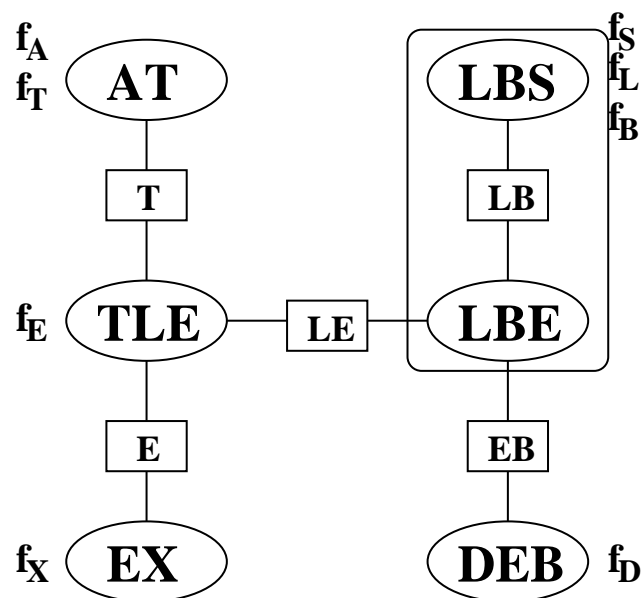
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia



- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$

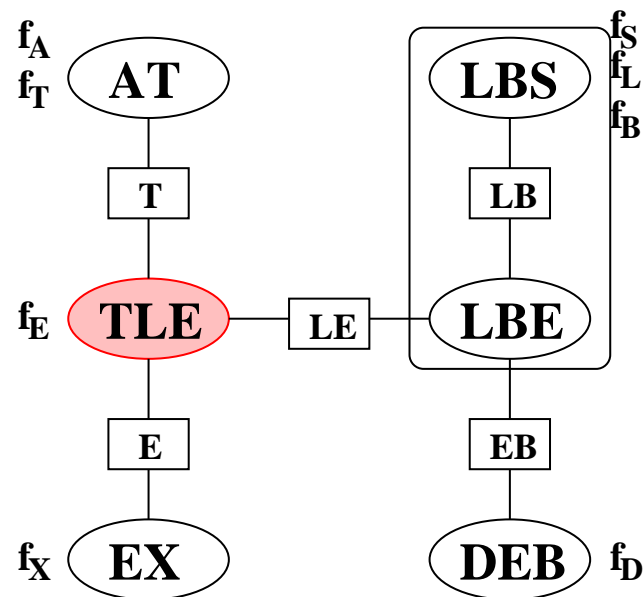
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$



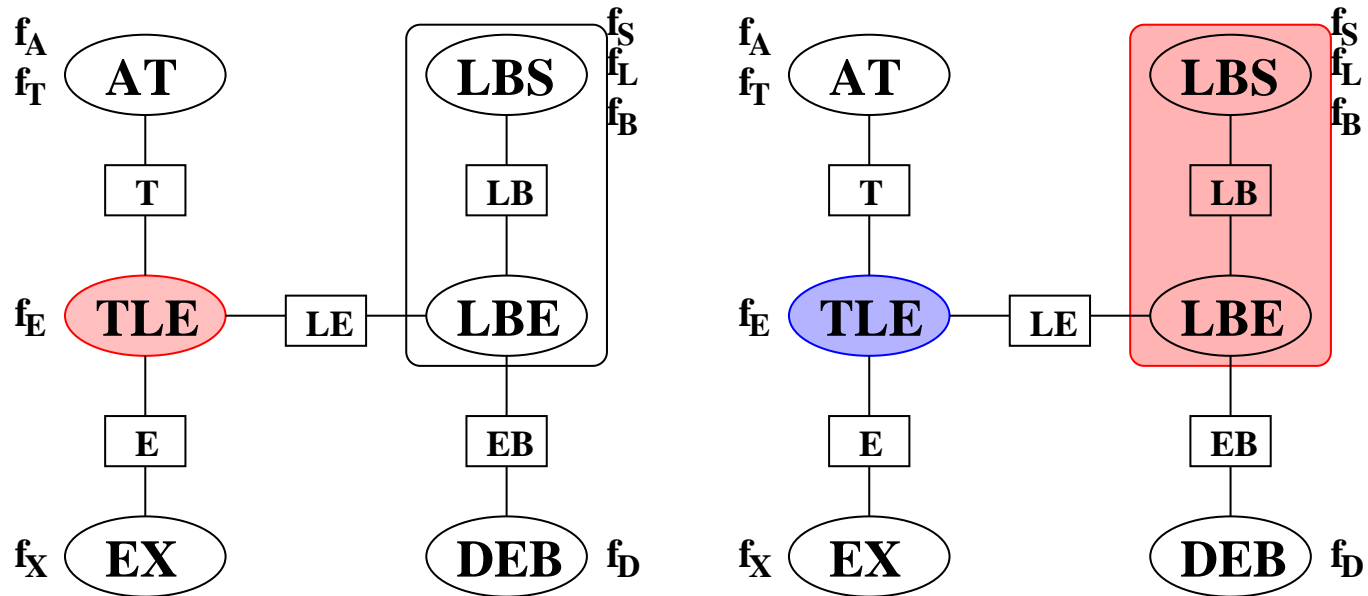
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$



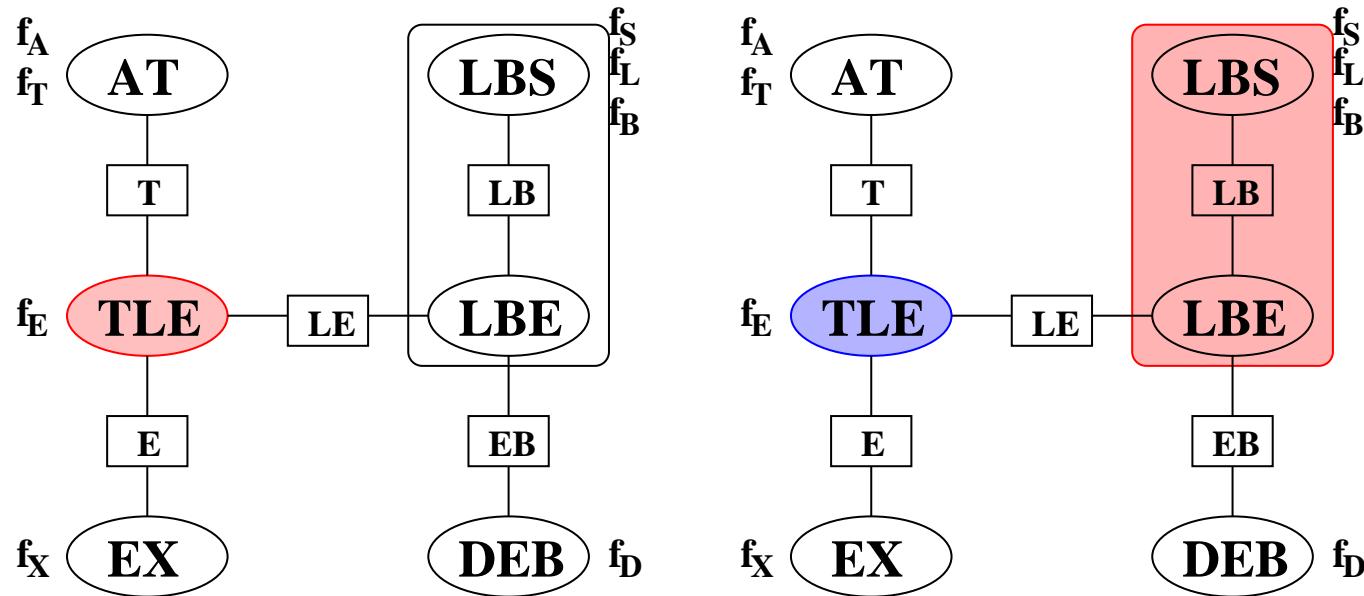
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$



Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

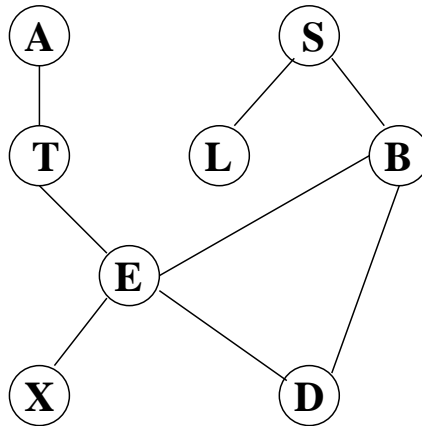
- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$



- Luego los subgrafos marcados son (TLE) y $(LBSE)$.

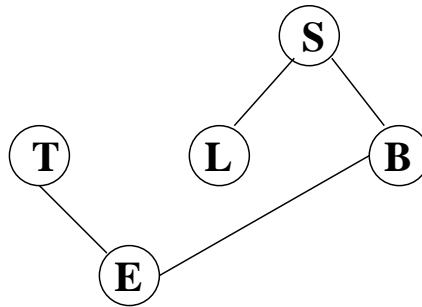
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



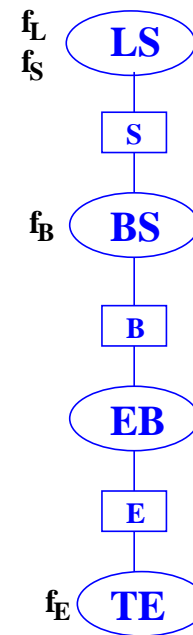
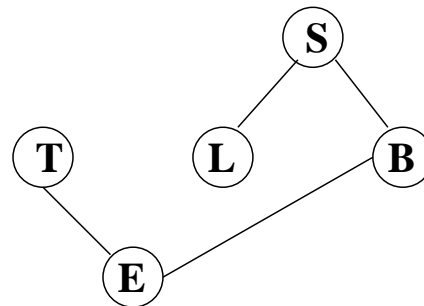
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



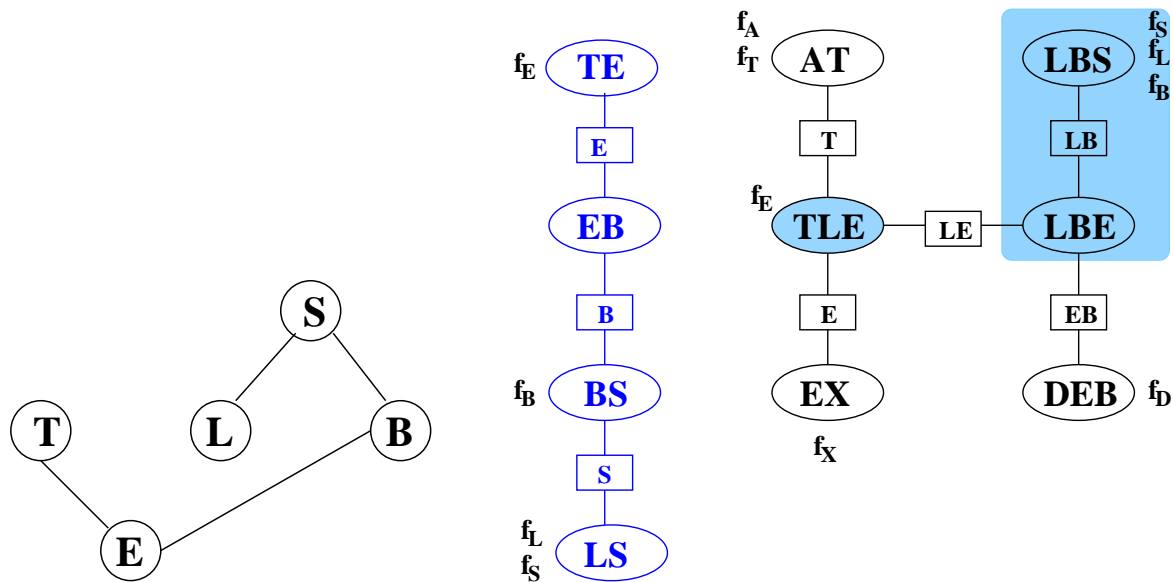
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



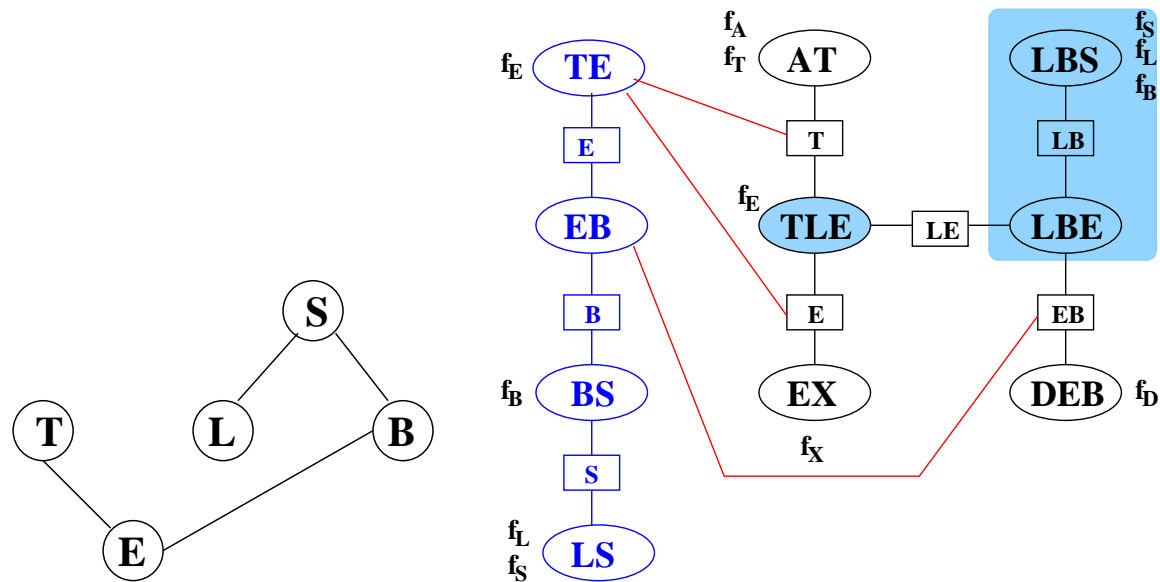
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



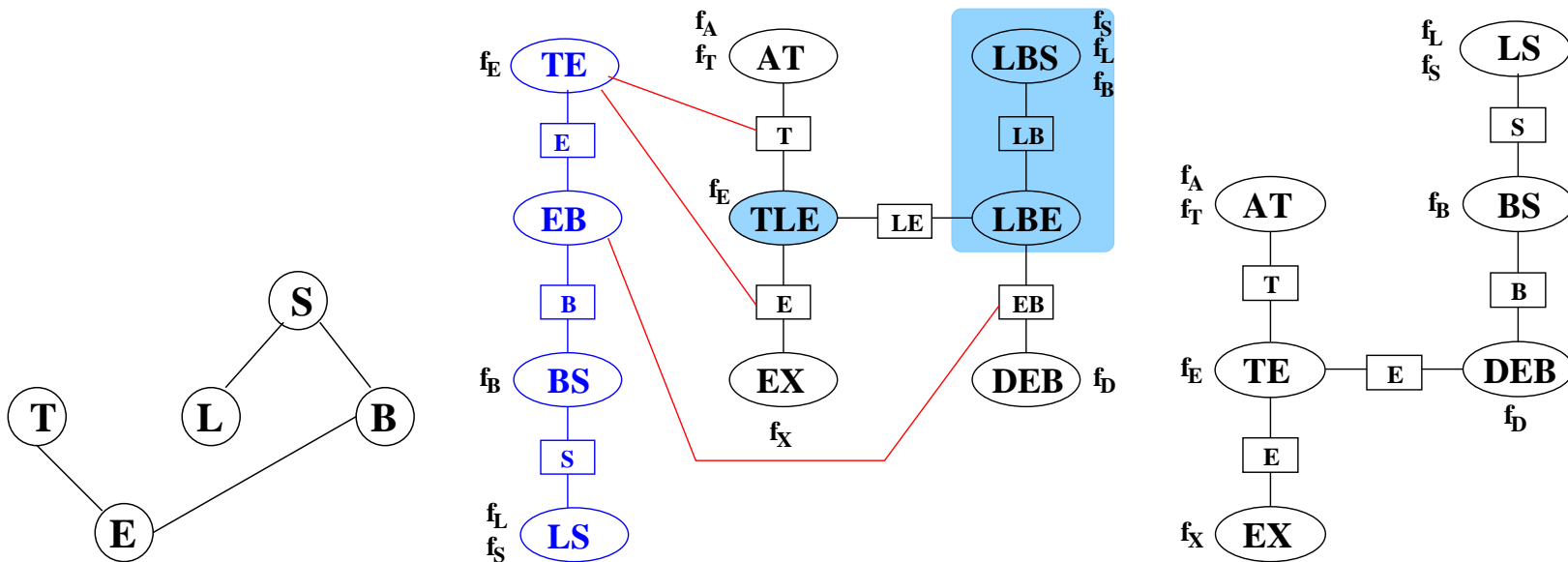
Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



Ejemplo: borrando el enlace $L \rightarrow E$ en Asia

- Luego la lista de enlaces eliminados es $L = \{(L, E), (T, L)\}$
- Luego los subgrafos marcados son (TLE) y $(LBSE)$.



- Hasta ahora trabajo metodológico

Conclusiones y trabajo futuro

- Hasta ahora trabajo metodológico
- Implementación en Elvira a falta de depurar

Conclusiones y trabajo futuro

- Hasta ahora trabajo metodológico
- Implementación en Elvira a falta de depurar
- Realizar una evaluación experimental

Conclusiones y trabajo futuro

- Hasta ahora trabajo metodológico
- Implementación en Elvira a falta de depurar
- Realizar una evaluación experimental
- Procesar de forma conjunta algunas modificaciones
⇒ heurísticas

Conclusiones y trabajo futuro

- Hasta ahora trabajo metodológico
- Implementación en Elvira a falta de depurar
- Realizar una evaluación experimental
- Procesar de forma conjunta algunas modificaciones
⇒ heurísticas
- ¿Integrar el proceso de CI en la interfaz de Elvira?

Hasta aquí
pescao vendido

