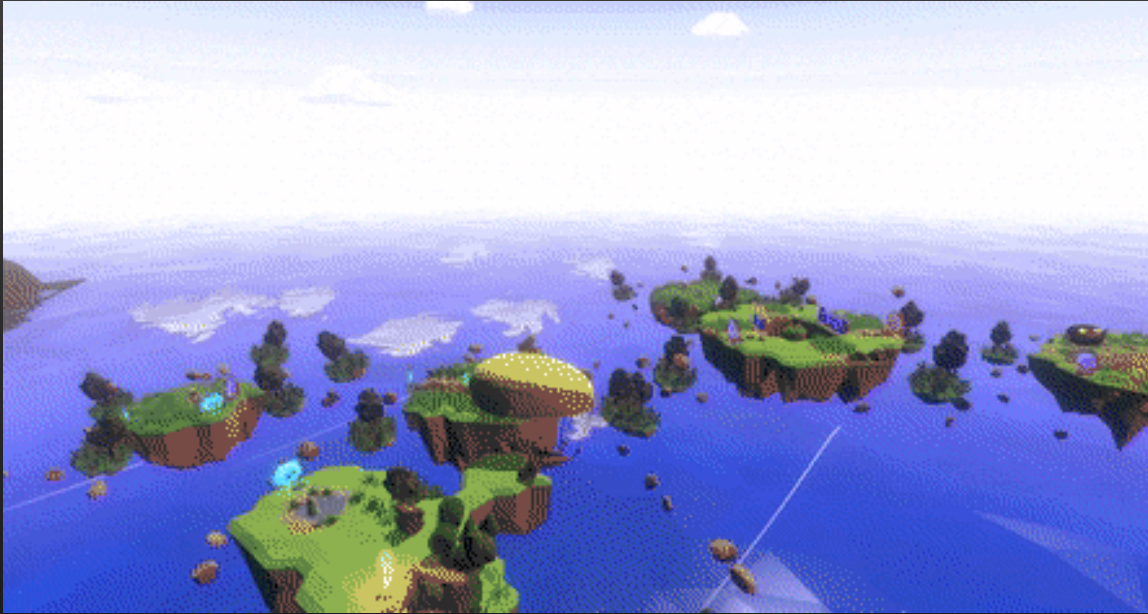


PROYECTO FINAL INTELIGENCIA ARTIFICIAL I

Minería de reglas de asociación espacial
sobre Minecraft

Martín Cogo Belver

MOTIVACIÓN



- Este proyecto tiene como objetivo la resolución de algún problema utilizando **algoritmos de Inteligencia Artificial**.
- Dentro de la industria de creación de videojuegos existe un área centrada en la generación procedural.
- Se le llama generación procedural de contenido a la creación algorítmica de contenido para videojuegos con entradas del usuario limitadas o indirectas.
- Existen varios algoritmos que permiten la generación de contenido a partir de parámetros de entrada que permiten al diseñador orientar el funcionamiento del algoritmo.

MOTIVACIÓN



- Teniendo en cuenta esta característica, surge la pregunta:
- *¿Sería posible mediante un algoritmo de inteligencia artificial y un entorno (artificial o natural), poder inferir parámetros para que un algoritmo de PCG genere entornos similares?*
- Se decidió que el tema del proyecto este en la búsqueda de algoritmos de inteligencia artificial que permitan **inferir** las relaciones que existen entre objetos en un espacio natural, artificial, físico o virtual.
- Como la obtención de datos de un entorno real, resulta complejo, se utiliza una porción de mapa del Videojuego **Minecraft**, el cual, utiliza la generación procedural para la creación de terrenos del juego utilizando cubos.
- Se eligió ya que los datos correspondientes a un terreno generado en el videojuego son: de fácil acceso y extracción, además, la disposición de los objetos dentro del juego son bloques con coordenadas rectangulares.

ENFOQUE



- Existe un área de investigación dentro de la ciencia de minería de datos que está enfocada en cómo realizar las tareas de minería de datos sobre bases de datos espaciales o geográficas.
- La **Minería de Datos Espaciales** o **SDM** es un proceso de descubrimiento y extracción de conocimiento generalizado sobre gran cantidad de datos espaciales.
- Analizando nuestro el mapa podemos notar que se encuentra totalmente generado por bloques de distinto tipo y cuyo posicionamiento cumple con un patrón que nosotros buscamos descubrir a lo largo del trabajo.

- Es importante distinguir que si se utilizara un algoritmo de **clustering**, los resultados que obtendremos son grupos de bloques que son similares y no es el tipo de respuesta que se busca para la solución al problema.
- Para la *búsqueda de patrones* en datos espaciales se tomó el enfoque basado en **reglas de asociación**. Este se concentra en la creación de transacciones booleanas sobre el espacio de tal manera que se pueda usar el algoritmo **Apriori**.
- Por lo tanto en este trabajo se generarán reglas de asociación con este algoritmo y se generarán gráficos que ilustren y permitan el análisis de resultados y obtención de una conclusión.

Métricas

Para evaluar las reglas utilizaremos las siguientes métricas:

- Soporte: Se trata de la probabilidad de que aparezca X e Y en las transacciones.

$$\text{Support}(X \rightarrow Y) = \frac{\text{Transaction containing X and Y}}{\text{Total number of transactions}}$$

- Confianza: Mide la probabilidad de que aparezca el Y dado que en una transacción aparece X.

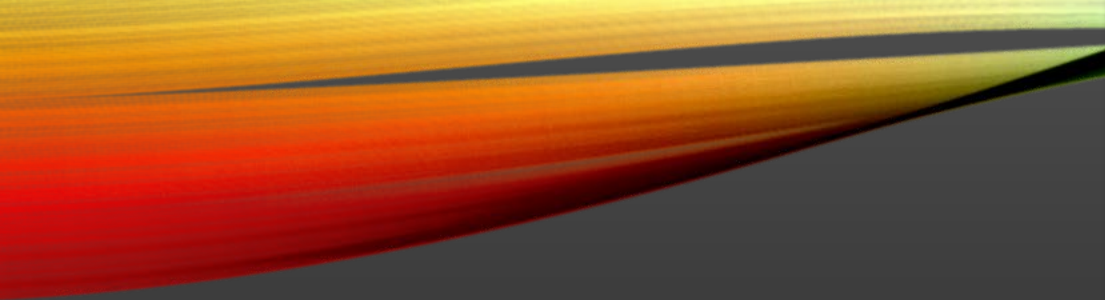
$$\text{Confidence}(X \rightarrow Y) = \frac{\text{Transaction containing X and Y}}{\text{Transactions containing X}}$$

RESULTADOS

Experimentos:

1. Windows centric model
 1. Transacciones con ventanas de 4x4x4.
2. Reference feature centric model
 1. Transacciones centradas en el atributo **diamond**. (Diamante)
 2. Transacciones centrada en el atributo **grass_block**. (bloques de pasto)
 3. Transacciones centradas en el atributo **tree_log** y **tree_leaves**. (troncos y hojas)
 4. Transacciones centradas en todos los atributos de un **chunk**. (todos los posibles bloques)





TRANSACCIONES DE VENTANAS DE 4X4X4 SIN SOLAPAMIENTO

- `tiempo de pre procesamiento` = 9 minutos
- `total de transacciones` = 20480
- `soporte mínimo` = $100/20480 = 0.0049$
- `confianza mínima` = 90%

Las transacciones implementan un **Window centirc model** y en este caso son ventanas de 4x4x4 es decir cubos de 64 bloques.

```
`tiempo de pre procesamiento` = 9 minutos
```

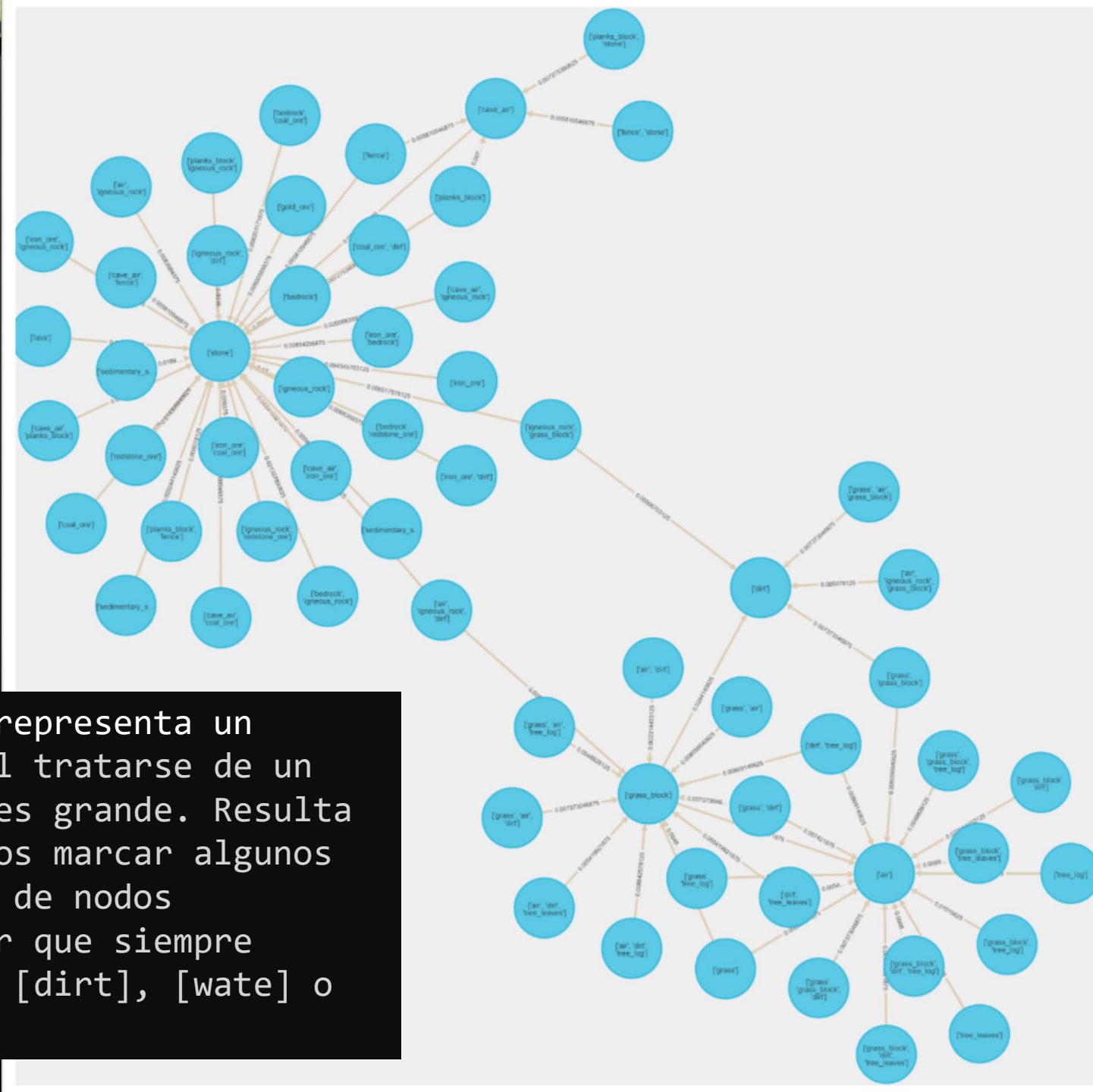
```
`total de transacciones` = 20480
```

```
`soporte mínimo` = 100/20480 = 0.0049
```

```
`confianza mínima` = 90%
```

Las transacciones implementan un **Window centirc model** y en este caso son ventanas de 4x4x4 es decir cubos de 64 bloques.

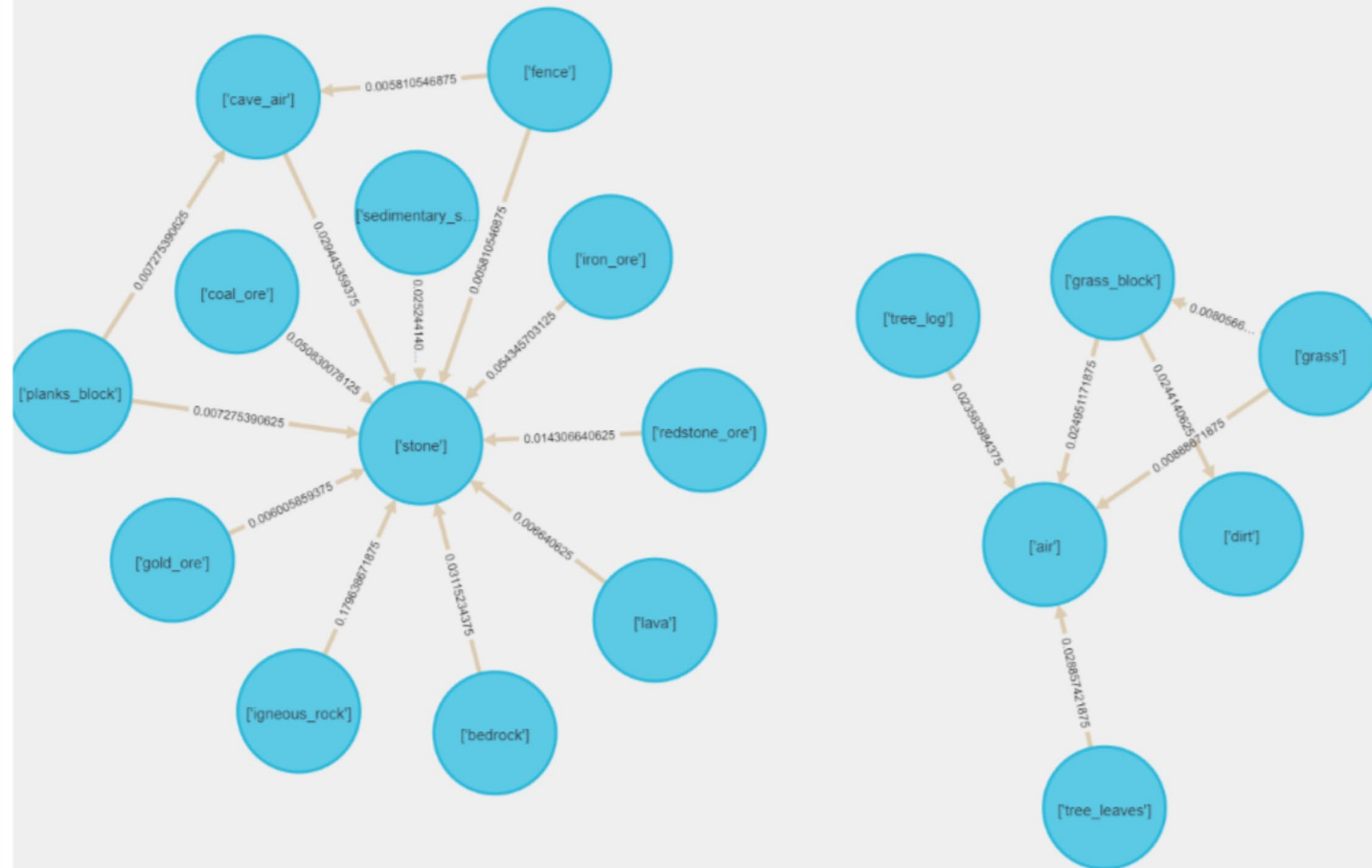
Grafo totalmente conexo, donde cada nodo representa un antecedente o consecuente de una regla. Al tratarse de un grafo muy grande y la cantidad de reglas es grande. Resulta muy difícil analizarlo. Simplemente podemos marcar algunos pequeños clusters que se forman alrededor de nodos particulares, los cuales se puede apreciar que siempre tienen un solo tipo de bloque, ej. [air], [dirt], [wate] o [Stone].



El grafo ilustra las reglas del conjunto DS que es el resultado de la poda de reglas realizada como post-procesamiento.

Estás son las reglas que representan y dan dirección a la mayoría de las demás reglas que se encuentran en non-DS rules.

El análisis resulta sencillo en este grafo.



En las reglas del conjunto DS la confianza es muy alta en pero el soporte no.

Esto se interpreta como que el consecuente ocurre muy frecuentemente cuando el antecedente se cumple pero ambos antecedente y antecedente ocurre poco frecuentemente juntos en la misma venta.

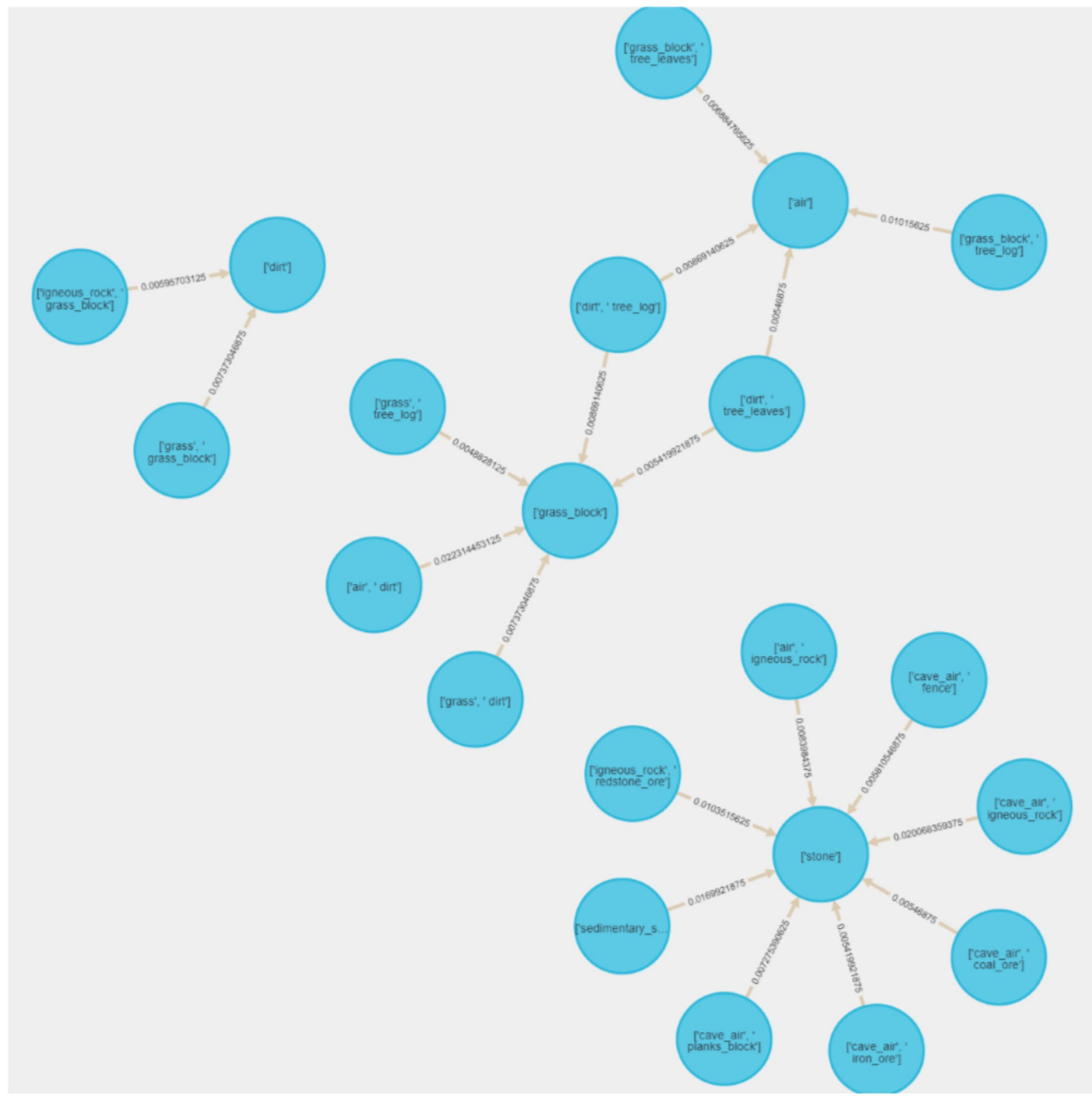
(De todas maneras el soporte es tan bajo debido a la enorme cantidad de bloques y lo desbalanceado que se encuentra el conjunto de bloques).

id	antecedants	=>	consequents	support	confidence
0	['coal_ore']	['=>']	['stone']	0.050830	1.000000
1	['lava']	['=>']	['stone']	0.006641	1.000000
2	['tree_leaves']	['=>']	['air']	0.028857	1.000000
3	['grass']	['=>']	['grass_block']	0.008057	0.906593
4	['grass_block']	['=>']	['dirt']	0.024414	0.902527
5	['iron_ore']	['=>']	['stone']	0.054346	1.000000
6	['sedimentary_stone']	['=>']	['stone']	0.025244	0.966355
7	['tree_log']	['=>']	['air']	0.023584	0.997934
8	['cave_air']	['=>']	['stone']	0.029443	0.982085
9	['grass_block']	['=>']	['air']	0.024951	0.922383
10	['planks_block']	['=>']	['stone']	0.007275	1.000000
11	['gold_ore']	['=>']	['stone']	0.006006	1.000000
12	['fence']	['=>']	['stone']	0.005811	1.000000
13	['fence']	['=>']	['cave_air']	0.005811	1.000000
14	['redstone_ore']	['=>']	['stone']	0.014307	1.000000
15	['igneous_rock']	['=>']	['stone']	0.179639	0.997019
16	['grass']	['=>']	['air']	0.008887	1.000000
17	['planks_block']	['=>']	['cave_air']	0.007275	1.000000
18	['bedrock']	['=>']	['stone']	0.031152	1.000000

Este grafo muestra el Conjunto no-DS que contiene todas las demás reglas que son combinaciones de todas las reglas en DS.

En este caso no nos revela mayor información que las reglas en DS.

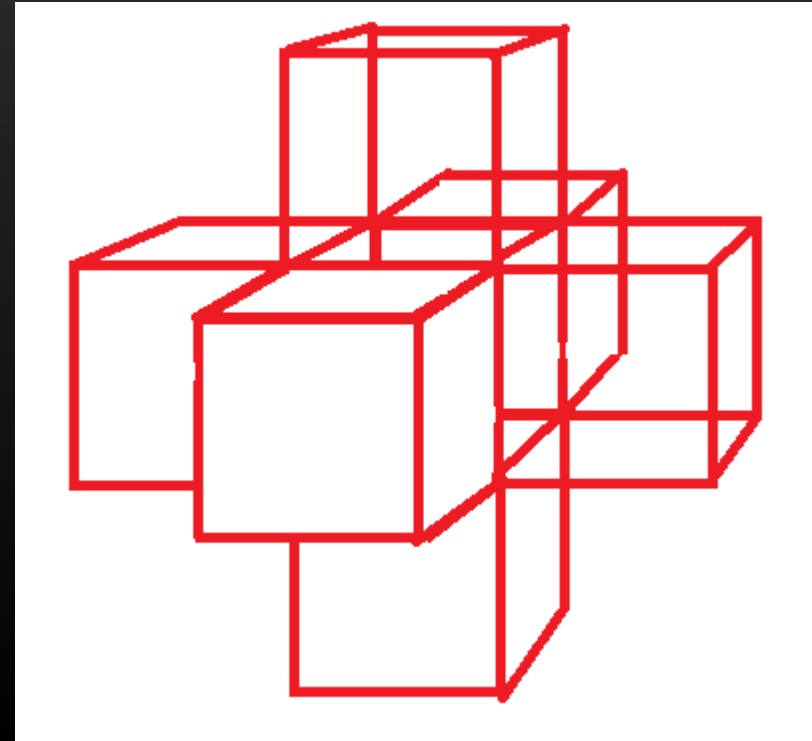
Cabe destacar que existen patrones que ocurren menos de 100 veces y que resultan importantes por ejemplo - Si hay **piedra** y **lava** y **agua** **entonces** hay **obsidiana**. Esta regla no fue obtenida pero puede verse claramente en el videojuego. Esto nos lleva a pensar que definir una cantidad mínima de veces para que aparezca un patrón no es tal vez la mejor forma de encarar el problema.



OTRO MODELO

Realmente este enfoque para modelar las transacciones no nos resulta de gran interés, ya que simplemente nos muestra que bloque existe en una ventana dado que otro existe en ella. Si bien es notorio que existe un patrón, no podemos saber claramente como es el patrón claramente. Entonces, el **windows centric model** no es un modelo que nos permita obtener las reglas que se buscan.

Los próximos experimentos se realizan con un **Reference feature centric model**. Las transacciones con ventanas de la forma cruz con 7 bloques, donde el bloque que hay en el centro es de un atributo determinado en el cual se centra el modelo.



Atributo de referencia diamante

`tiempo de preprocesamiento` = 9 segundos

`total de transacciones` = 75

`soporte mínimo` = $12/75 = 0.16$

`confianza mínima` = 75%

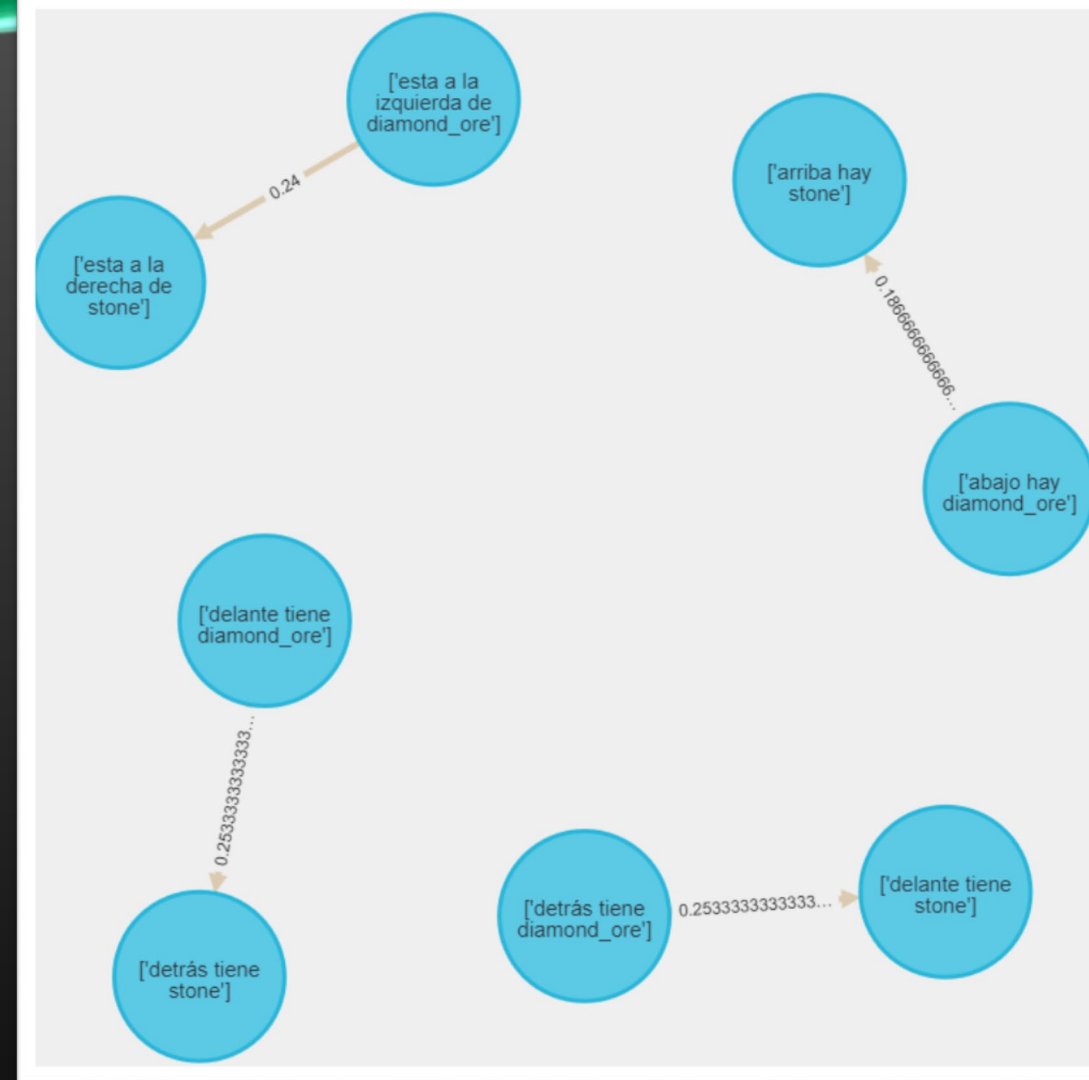
Para empezar no tenemos un nodo con ítem [es diamond_ore]. Esto se debe principalmente a que este ítem aparece en todas las transacciones generadas en el pre-procesamiento.

En toda regla donde [es diamond_ore] haya aparecido tiene independencia estadística, ya que la aparición de [es diamond_ore] en conjunto de transacciones no depende de ningún otro Atributo.

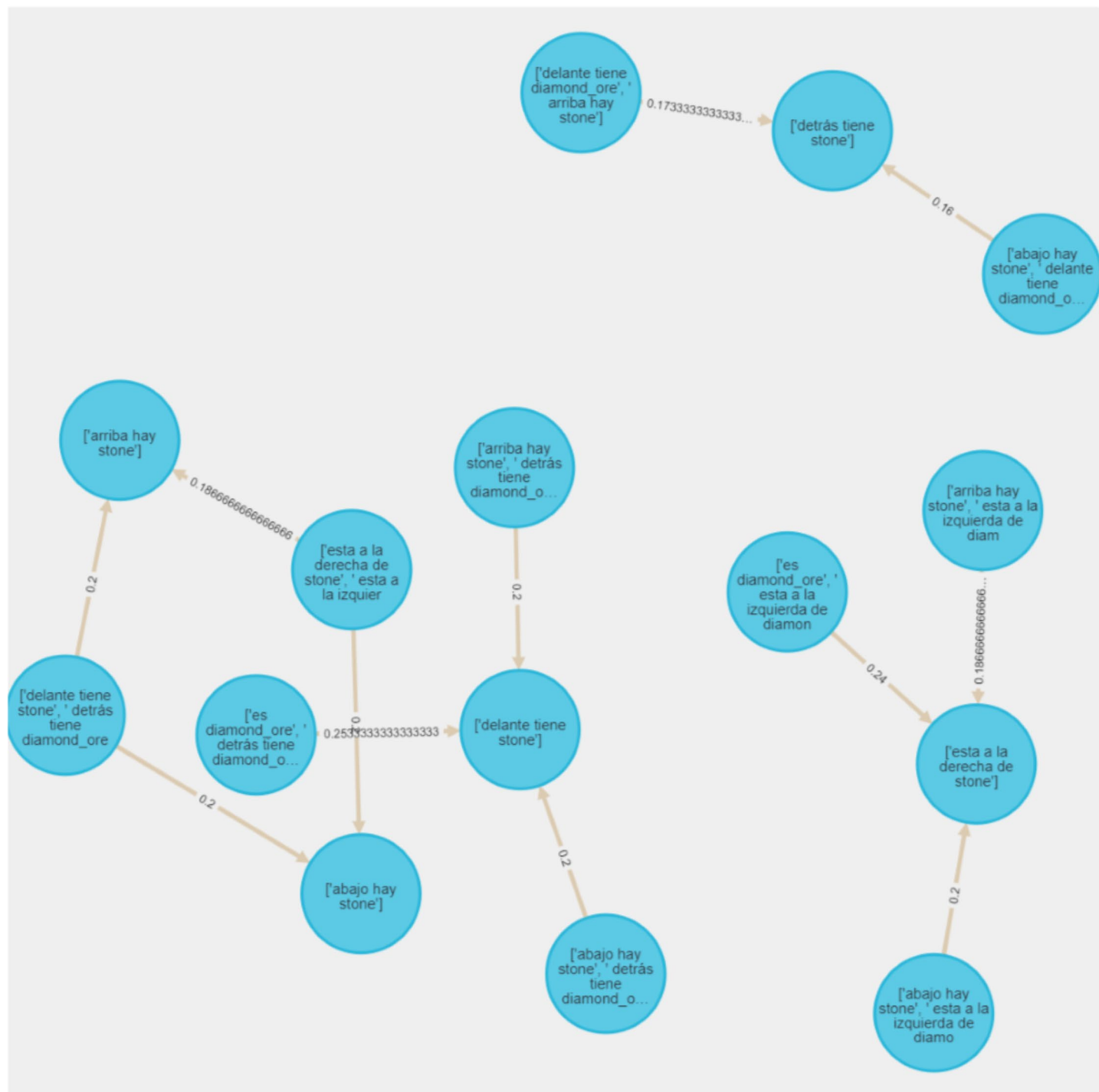
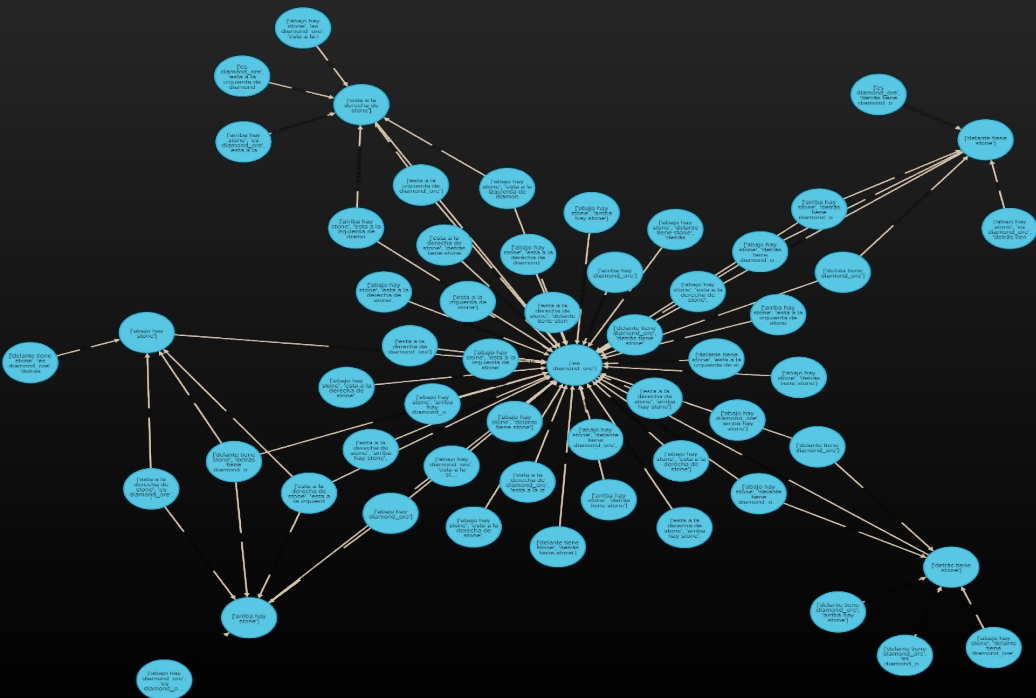
En el conjunto de reglas DS solo tenemos 4 presentes y todas ilustran un patrón que se podría expresar coloquialmente como:

Los bloques de diamante en el juego están rodeados de bloques de piedra.

	antecedants	=>	consequents	support	confidence
0	['detrás tiene diamond_ore']	['=>']	['delante tiene stone']	0.253333	0.791667
1	['abajo hay diamond_ore']	['=>']	['arriba hay stone']	0.186667	0.823529
2	['está a la izquierda de diamond_ore']	['=>']	['está a la derecha de stone']	0.240000	0.857143
3	['delante tiene diamond_ore']	['=>']	['detrás tiene stone']	0.253333	0.791667



- Una vez que hemos reconocido este patrón si miramos las reglas del conjunto no-DS en la no encontraremos reglas que describan otro patrón distinto que el que ya se ha mencionado.
- A simple vista podemos notar que todas las reglas generadas es una gran grafo difícil de analizar.

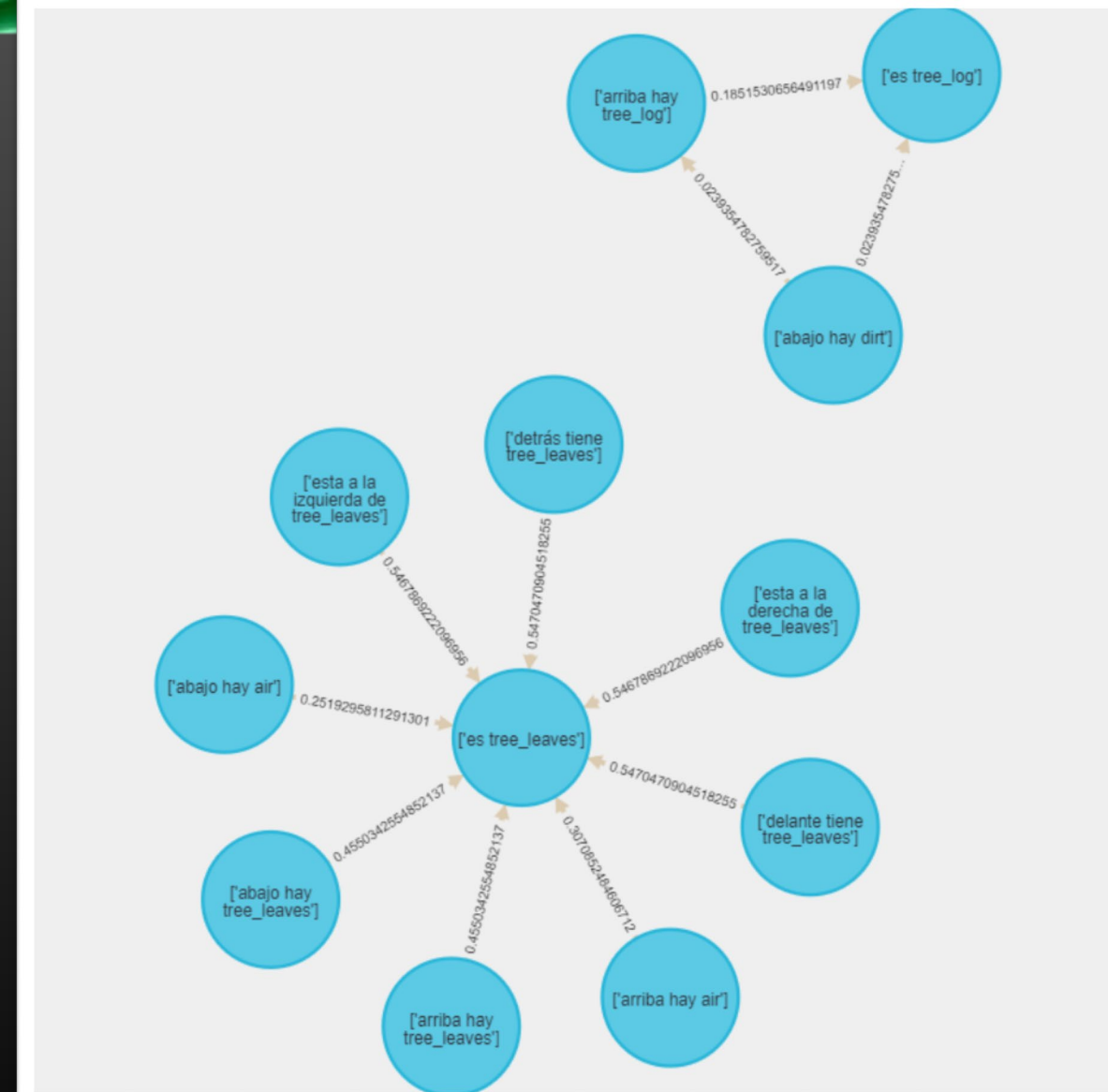


Atributo de referencia ****hojas y troncos****

`tiempo de preprocesamiento` = 24 minutos
`total de transacciones` = 11531
`soporte mínimo` = $200/11531 = 0.0173$
`confianza mínima` = 90%

Observando el Grafo del conjunto DS se ven claro los patrones en este conjunto:

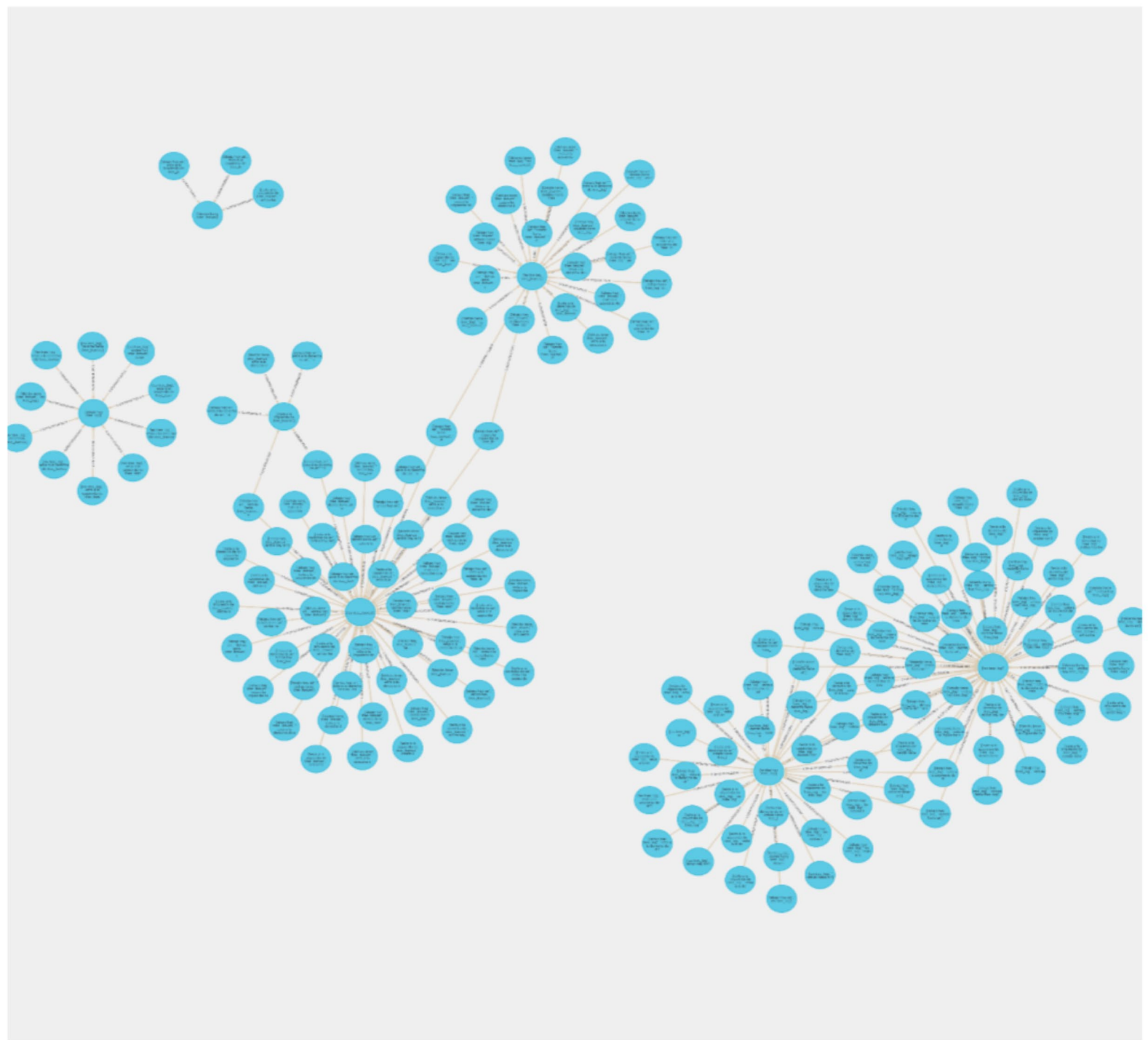
- El patrón que indica que si hay un bloque de tierra abajo se trata de un tronco y arriba tiene otro tronco. (Corresponde con el tallo del árbol)
- El patrón que indica que si hay una bloque de tipo hoja en el juego este está rodeada de bloques de hoja y por arriba o debajo puede haber bloques de tipo aire. (Corresponde con la copa del árbol)



Observando el grafo de reglas no-DS hay clusters alrededor de los ítems [arriba hay tree_leaves], [es tree_leaves], [abajo hay tree_log], [es tree_log] y [arriba hay tree_log].

Aunque es engorroso de analizar el patrón mas importante que puede inferirse es:

- Los bloques de tipo tronco pueden estar rodeados de bloques de aire o rodeados por bloques de hoja o bloques de tronco.



Atributo de referencia pasto

`tiempo de preprocesamiento` = 6 minutos
`total de transacciones` = 4108
`soporte mínimo` = $200/4108 = 0,0486$
`confianza mínima` = 90%

Del grafo del conjunto DS se puede ver que las reglas tienden a relacionarse con 3 ítems en particular son: [abajo hay dirt] y [arriba hay air].

Esto muestra un claro patrón:

- ****Si el bloque es de tipo pasto, debajo tiene bloques de tierra y por arriba tiene bloques de aire.****
- ****Si el bloque es de tipo pasto puede estar rodeado de bloques de tierra o más bloques de pasto****

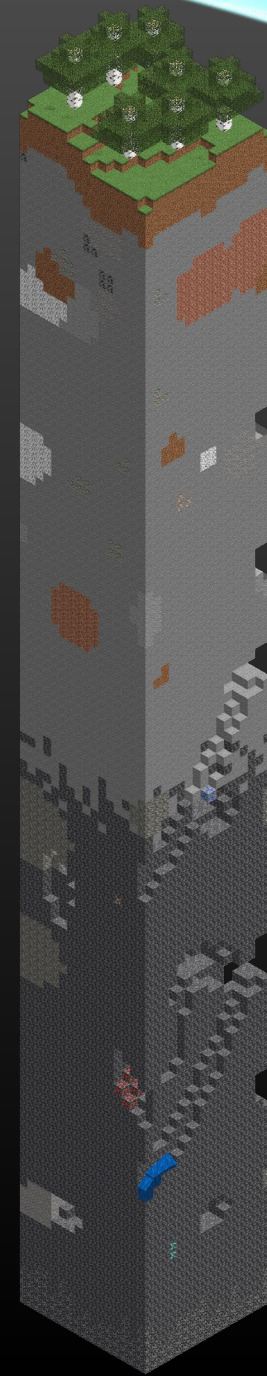
También, podemos decir que el patrón donde arriba de un grass_block aparecen flores tiene una frecuencia muy pequeña como para que aparezca en las reglas que hemos generado, pero aún así se trata de un patrón totalmente válido y no descubierto. Esto nos hace replantear si realmente estamos tomando un acercamiento correcto al problema.



Con los resultados obtenidos mediante los dos modelos anteriores, podría querer combinar la forma que toman las reglas como resultado de el **reference feature centric model** y aplicarlo en todo el espacio como con el **window centric model**.

Debido a que la cantidad de tiempo necesario para generar las ventanas en cruz para cada bloque del mapa es grande, nos dedicaremos a el análisis y exploración de las reglas generadas para un solo chunk que son un total de $16 \times 16 \times 256 = 65536$ bloques.

Por lo tanto generamos las transacciones para todos los bloques con ventanas tipo cruz que se superpondrán en todo el espacio.



Combinación entre los modelos anteriores

`tiempo de pre procesamiento` = 13 minutos y 30 segundos

`total de transacciones` = 65536

`soporte mínimo` = $200/65536 = 0.0003$

`confianza mínima` = 90%

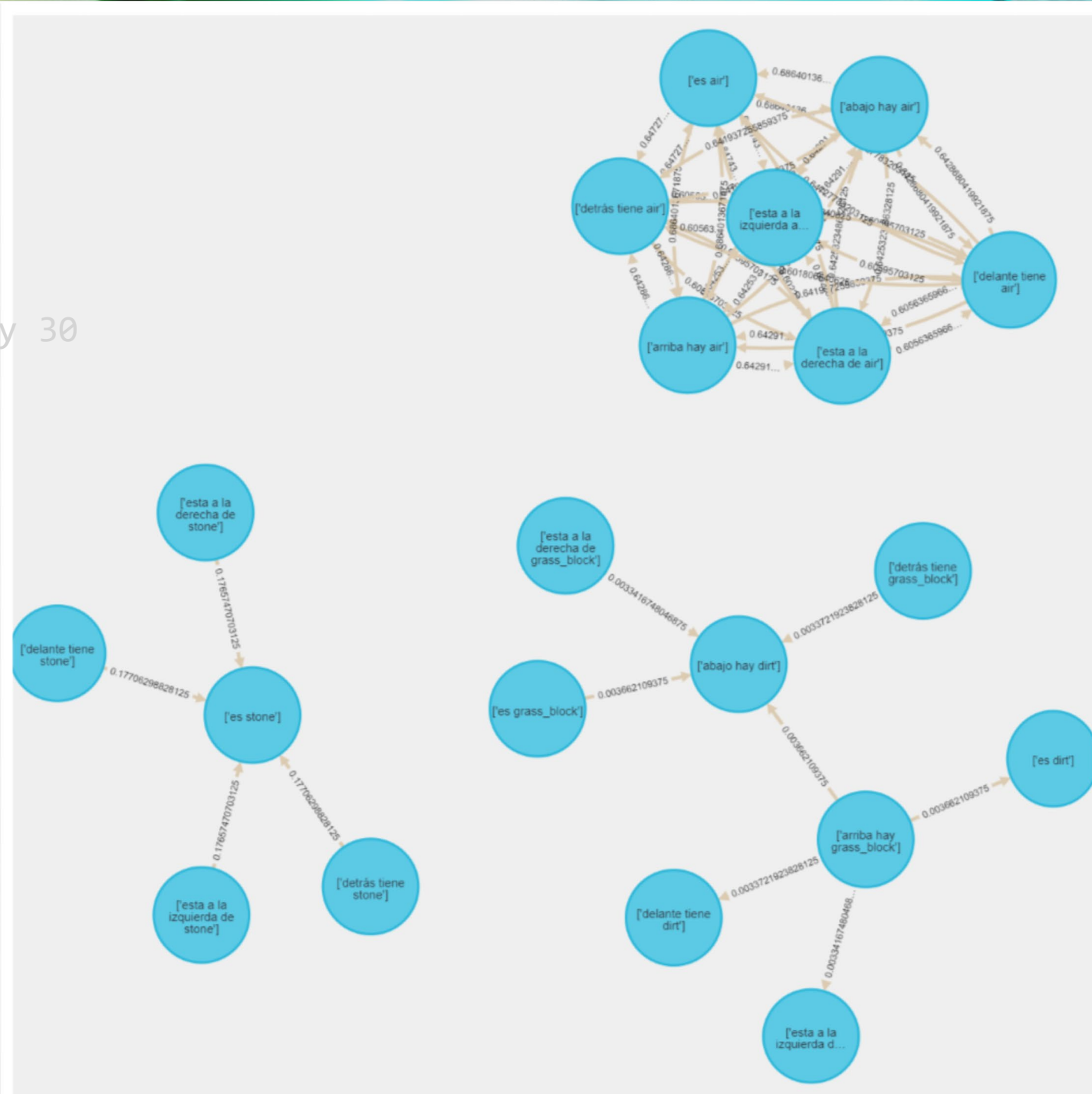
En este caso mirando el grafo se mantienen 3 clusters que muestran los siguientes patrones:

- Si el bloque es de tipo aire, tiende a estar rodeado de otros bloques de tipo aire.

Este patrón aparece una enorme cantidad de veces y por eso ese cluster (En el grafo la parte superior derecha) se puede ver densamente conectado.

- Si el bloque es de tipo piedra, está rodeado de bloques de tipo piedra

- Si el bloque es de tierra, arriba hay bloques de pasto y abajo bloques de tierra



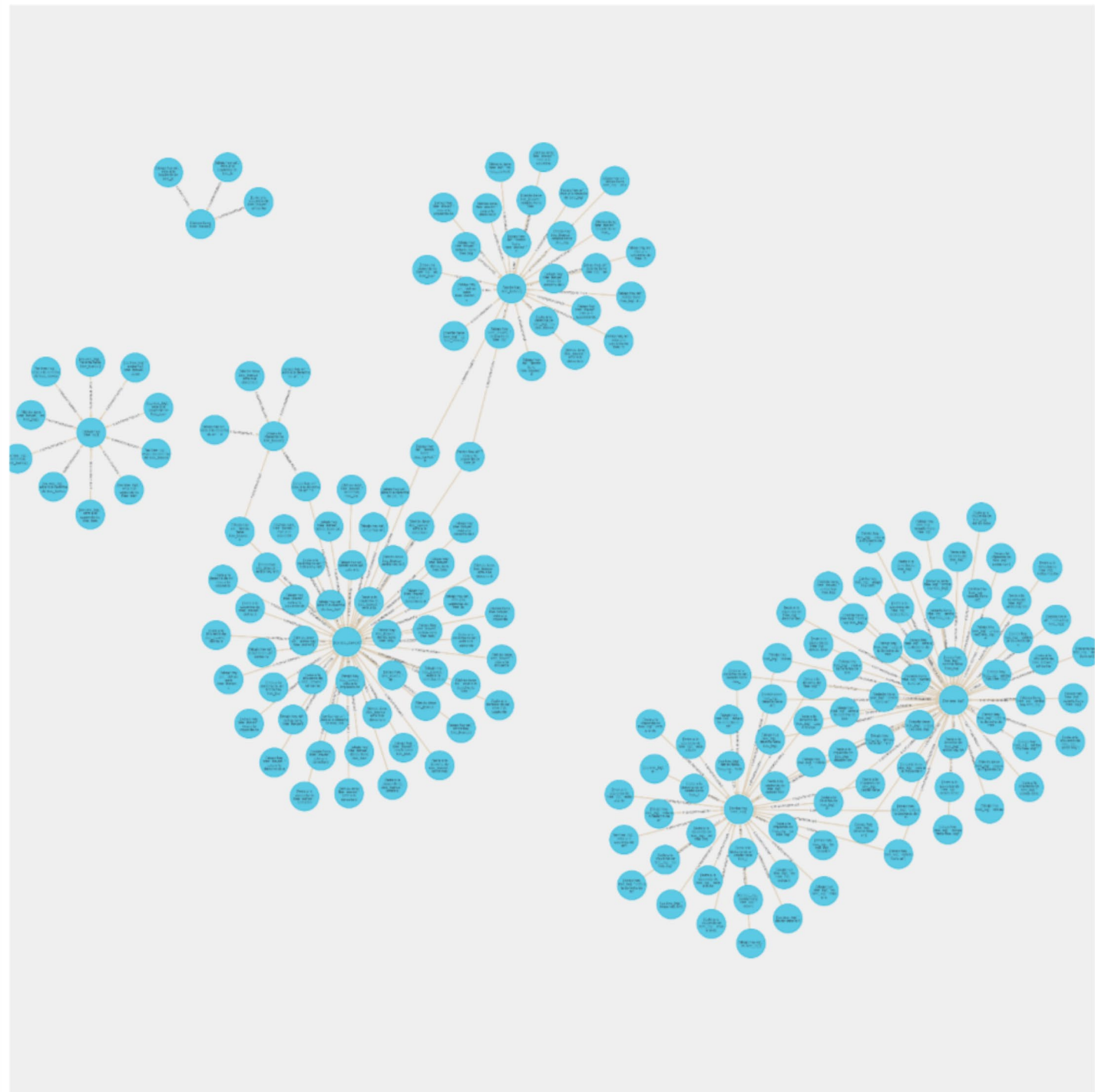
Esta información no es novedosa y explica muy poco respecto a todos los patrones que existen en un chunk del mapa Minecraft.

Esto se debe a que los patrones descubiertos son los más frecuentes que se presentan en el **chunk**.

Buscar patrones con menor soporte implicaría la generación de mayores cantidad de reglas redundantes y poco útiles, entorpeciendo el la búsqueda de patrones.

Mirando, hay cluster donde se relacionan los ítems con bloques de aire, por otro lado los que se relacionan con bloques de piedra, un cluster relacionado a las `igneous_rocks`(arena, arcilla o grava).

Interpretar patrones de este conjunto resulta en una tarea más ardua.



CONCLUSIÓN

- Si bien podemos adaptar nuestro problema para utilizar **Apriori**, no termina siendo lo más conveniente. Sobre todo porque nos encontramos con limitados relacionadas con el **tamaño de las ventanas**, y el valor del **soporte mínimo**.
 - Definir **ventanas** para elaborar las transacciones no resulta complejo, pero la cantidad de tiempo requerida en esta fase de pre procesamiento es alto comparada con el tiempo necesario para la generación de reglas y depende de **la cantidad de bloques** que se usen en la entrada. Y además, dependiendo del problema y la forma que se desee que tengan las reglas de salida, más difícil resulta generar estas ventanas.
 - Con respecto al **soporte mínimo**, cómo el conjunto de datos está muy **desbalanceado**, si buscamos conseguir los patrones que son poco frecuentes, la forma de obtenerlos es bajando el *soporte mínimo* lo cual lleva a la generación de una gran cantidad de reglas donde muchas son **redundantes** o **pequeñas variaciones de aquellas reglas con mayor soporte**.
- Para lidiar con la gran cantidad de reglas generadas, utilizamos el algoritmo **P-DS** para la poda de reglas. Si bien, el algoritmo devuelve un conjunto de reglas que podrían servir de resumen, tiende a estar formado por todas aquellas reglas con **un solo antecedente y consecuente**. Las reglas del conjunto **no-DS** pueden formarse con las reglas de **DS** pero, **no es sencillo determinar si existen** más patrones respecto a los ya encontrados en **DS**, sobre todo por el tamaño de **no-DS**.

En los resultados la gran cantidad de patrones que existen en la porción de mapa de Minecraft **han resultado difíciles de obtener**, y no solo eso, el **análisis de los resultados no es sencillo** para la evaluación del rendimiento de las pruebas.



Además, el hecho de que los patrones tengan que **ser deducidos por un experto**, indica que es necesario continuar en la búsqueda de un mecanismo de minería de reglas que expresen mejor los patrones que buscamos.

Debido a todas las complicaciones que se han presentado durante el proyecto **se concluye que la búsqueda de otro tipo de acercamiento** es necesario para solucionar el problema empleando inteligencia artificial y hallar otro algoritmo para la generación de reglas que sea más conveniente, aunque esto requeriría una exploración bibliográfica de algoritmos más complejos.

Si se siguiera con el enfoque de la minería de **reglas de asociación** y la utilización del algoritmo **Apriori**. El modelo para pre procesamiento y generación de transacciones sería el ***centric feature model***.

También sería necesario solucionar los 2 principales problemas que se observaron:

1. La búsqueda de representación de las reglas que faciliten el análisis de los resultados
2. Un método que permita marcar de manera efectiva aquellas reglas que sí son interesantes para el problema.

De las cosas que quedaron de manera tentativa y no se realizaron en el proyecto podemos mencionar:

- La implementación de un generador de representaciones gráficas con grafos. En este caso se utilizó la aplicación de escritorio de la base orientada a grafos **Neo4j** pero tuve que generar los gráficos a mano.
- Implementación de algún método adecuado que permita filtrar y minar reglas que involucren distintos tipos de bloque sin generar tantas reglas redundantes o que brinden información insignificante.
- Aplicar técnicas para trabajar con conjuntos de datos desbalanceados.



La implementación de un **cloud model** para las coordenadas rectangulares de los bloques, que con algún tipo de ventana específico podríamos generar reglas como por ejemplo:

[hay muchos bloques de tipo piedra en lo profundo]

-->

[hay algunos bloques de tipo oro en lo profundo]

Donde “muchos”, “pocos” y “algunos” se obtienen a partir de un **cloud model** para clasificar cantidad de un tipo de bloque y “profundo” puede ser el resultados de aplicar un **cloud model** sobre las coordenadas rectangulares de los bloques. Y el **Cloud model** debido a que nos permite de alguna manera tomar el dominio de las coordenadas rectangulares y generar expresarlo en lenguaje natural.



GRACIAS