

1. Opis projektu

1.1 Cel

Głównym celem projektu było stworzenie sieci bayesowskiej na podstawie danych o lasach. Analizowane były zależności między typem gruntu, a występowaniem istotnego zadrzewienia danym gatunkiem. Dodatkowymi czynnikami były: rodzaj gleby, funkcja lasu, wariant uwilgocenia gleby, typ siedliska.

1.2 Użyte narzędzia

Do uporządkowania danych został użyty JupyterLab, a do stworzenia sieci bayesowskich – GeNIE.

1.3 Dane

Dane zostały pobrane ze strony <https://www.bdl.lasy.gov.pl/portal/> dla nadleśnictwa Płaska z 2021 roku. Do analizy zostały wykorzystane pliki: f_storey_species.txt, f_tree_species_dic.txt, f_subarea.txt, f_soil_subtype_dic.txt.

1.4 Opis danych

Plik f_storey_species.txt zawiera opis gatunków w warstwie. Plik f_tree_species_dic.txt jest słownikiem gatunków drzew. Plik f_subarea.txt zawiera opis wydzielień. Plik f_soil_subtype_dic.txt jest słownikiem typów i podtypów gleb.

2. Porządkowanie danych

W celu zwiększenia czytelności nazw gatunków, przyłączam do tabeli zawierającej opis gatunków w warstwie kolumnę z nazwą gatunku z pliku ze słownikiem gatunków drzew.

```
[2]: storey_species = pd.read_csv("f_storey_species.txt", sep = "\t")
[3]: species = pd.read_csv("f_tree_species_dic.txt", sep = "\t")
[4]: storey_species = pd.merge(storey_species, species[['species_cd','species_name']], on = 'species_cd', how = 'left')
```

Tabela wynikowa - `storey_species` - ma 66943 wierszy oraz 12 kolumn.

Do analizy wybieram tylko te gatunki drzew, dla których w danym nadleśnictwie została wyliczona miąższość.

```
[6]: df = storey_species[~storey_species['volume'].isna()]
```

W wyniku otrzymuję tabelę `df`, która ma 17224 wierszy i 12 kolumn.

Aby sprawdzić, jakie gatunki występują w nadleśnictwie, grupuję tabelę po kodzie.

```
[8]: species_groups = df.groupby('species_cd')
```

Okazuje się, że jest 21 takich gatunków:

BK	buk pospolity	KSZ	kasztanowiec biały
BRZ	brzoza brodawkowata	LP	lipa drobnolistna
CZR	czereśnia pospolita	MD	modrzew europejski
DB	dąb nieokreślony	OL	olsza czarna
GB	grab pospolity	OS	topola osika
GR	grusza pospolita	SO	sosna zwyczajna
IWA	wierzba iwa	TP	topola biała
JB	jabłoń dzika	WB	wierzba biała
JS	jesion wyniosły	WIŚ	wiśnia pospolita
KL	klon pospolity	WZ	wiąz pospolity
		ŚW	świerk pospolity

Następnie tworzę tabelę przestawną, w której kolumny przedstawiają każdy gatunek, a wiersze – pojedyncze powierzchnie. Wartościami tabeli jest informacja, czy dany gatunek na danej powierzchni ma wyliczoną miąższość.

Pierwszym krokiem jest przestawienie tabeli.

```
[10]: df = df.pivot_table(index = 'arodes_int_num', columns = 'species_name', values = 'volume', aggfunc = 'sum')
```

Używam tu funkcji sumującej, ponieważ na jednej powierzchni może znajdować się więcej, niż jedno drzewo danego gatunku.

Tabela wynikowa ma 6058 wierszy oraz 21 kolumn. Aby zmniejszyć liczbę kolumn, łączę topolę białą z topolą osiką oraz wierzbę białą z wierzbą iwaną. Wartości NaN zastępuję zerami.

```
[14]: topola = df['topola biala'].fillna(0) + df['topola osika'].fillna(0)
wierzba = df['wierzba biala'].fillna(0) + df['wierzba iwa'].fillna(0)
```

```
[15]: df = df.drop(columns = ['topola biala', 'topola osika', 'wierzba biala', 'wierzba iwa'])
```

```
[16]: df['topola'] = topola
df['wierzba'] = wierzba
```

W wyniku otrzymuję 19 kolumn.

Następnym krokiem jest zamiana wartości w tabeli na zmienną dwuwartościową.

```
[18]: df = df.replace(0, np.nan)
```

```
[19]: df2 = df.notnull()
```

Wszystkie zera zostały zamienione na wartość NaN, a tabela df2 zawiera wartości True/False odpowiadające na pytanie czy jest wyliczona miąższość czy nie ma wyliczonej miąższości.

Do powstałej tabeli przyłączam kolumny z tabeli subarea mówiące o typie i podtypie gleby, funkcji lasu, uwilgoceniu gleby oraz typie siedliskowym lasu.

```
[21]: subarea = pd.read_csv("f_subarea.txt", sep = "\t")
```

```
[24]: df2 = pd.merge(df2, subarea[['arodes_int_num', 'soil_subtype_cd', 'forest_func_cd', 'moisture_cd', 'site_type_cd']], on='arodes_int_num', how='left')
```

Na podstawie kolumny site_type_cd mogę określić, czy siedlisko to bór, las, czy ols. Typy w tej kolumnie są zakodowane w pierwszej literze skrótu: B – bór, L – las, O – ols.

```
[26]: df2[['site_type_cd']] = df2[['site_type_cd']].replace(to_replace = r'\bB\b+', value = 'bór', regex = True)
df2[['site_type_cd']] = df2[['site_type_cd']].replace(to_replace = r'\bL\b+', value = 'las', regex = True)
df2[['site_type_cd']] = df2[['site_type_cd']].replace(to_replace = r'\bO\b+', value = 'ols', regex = True)
```

Dzięki słownikowi typów i podtypów gleb mogę je pogrupować typami. W tabeli df2 znajduje się 25 typów i podtypów gleb. Wyodrębniam je i łączę ze słownikiem.

```
[27]: soil_groups = df2.groupby('soil_subtype_cd').size().to_frame(name = 'count').reset_index()
[29]: soils = pd.read_csv("f_soil_subtype_dic.txt", sep = "\t")
[30]: soil_groups = pd.merge(soil_groups, soils[['soil_subtype_cd', 'soil_subtype_name']], on = 'soil_subtype_cd', how = 'left')
```

	soil_subtype_cd	count	soil_subtype_name				
0	B	1	Gleby bielicowe ...	13	MRm	107	Gleby mineralno-murszowe ...
1	BR	33	Gleby brunatne ...	14	MRms	65	Gleby murszaste ...
2	BRwy	16	Gleby brunatne wyługowane ...	15	MRw	119	Gleby murszowe właściwe ...
3	Bgms	54	Gleby glejo-bielicowe murszaste ...	16	Mt	223	Gleby torfowo-murszowe ...
4	Bgts	37	Gleby glejo-bielicowe torfiaste ...	17	OGw	3	Gleby opadowoglejowe właściwe ...
5	Bgw	686	Gleby glejo-bielicowe właściwe ...	18	Pbr	5	Gleby płowe brunatne ...
6	Bw	582	Gleby bielicowe właściwe ...	19	RDb	1407	Gleby rdzawe bielicowe ...
7	CZms	20	Czarne ziemie murszaste ...	20	RDbr	97	Gleby rdzawe brunatne ...
8	CZw	2	Czarne ziemie właściwe ...	21	RDw	1327	Gleby rdzawe właściwe ...
9	Dw	1	Gleby deluwialne właściwe ...	22	Tn	561	Gleby torfowe torfowisk niskich ...
10	Gt	59	Gleby gruntowoglejowe torfowe ...	23	Tp	26	Gleby torfowe torfowisk przejściowych ...
11	Gts	10	Gleby gruntowoglejowe torfiaste ...	24	Tw	59	Gleby torfowe torfowisk wysokich ...
12	Gw	117	Gleby gruntowoglejowe właściwe ...				

Na tej podstawie, przypisuję typy gleb.

```
[33]: df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('B', 'bielicowe')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('Bw', 'bielicowe')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('BR', 'brunatne')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bBR\w+', value = 'brunatne', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bBg\w+', value = 'glejo_bielicowe', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bCZ\w+', value = 'czarne_ziemie', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\BD\w+', value = 'deluwialne', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bG\w+', value = 'gruntowoglejowe', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('MRm', 'mineralno_murszowe')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('MRms', 'murszaste')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('MRw', 'murszowe')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace('Mt', 'torfowo_murszowe')
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bGw\w+', value = 'opadowoglejowe', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bP\w+', value = 'płowe', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bRD\w+', value = 'rdzawe', regex = True)
df2[['soil_subtype_cd']] = df2[['soil_subtype_cd']].replace(to_replace = r'\bT\w+', value = 'torfowe', regex = True)
```

Zmieniam nazwy kolumn

```
[36]: df2.rename(columns = {'soil_subtype_cd':'gleba', 'forest_func_cd':'funkcja_lasu',
                           'moisture_cd':'uwilgotnienie', 'site_type_cd':'siedlisko' }, inplace = True)
```

Sprawdzam, czy kolumny nie mają jednej wartości we wszystkich wierszach.

gleba	funkcja_lasu	uwilgotnienie	siedlisko
rdzawe	OCHR	Ś	bór
rdzawe	OCHR	Ś	bór
rdzawe	OCHR	Ś	bór
rdzawe	OCHR	Ś	bór
rdzawe	OCHR	Ś	bór
...
NaN	NaN	NaN	NaN
rdzawe	OCHR	Ś	bór
NaN	GOSP	Ś	bór
glejo_bielicowe	OCHR	WW	las
NaN	GOSP	BM	las

Po samym wypisaniu tabeli, widać, że wartości się powtarzają.

Upewniam się, że dane nie zawierają „pustych” kolumn (z samymi wartościami False).

```
[36]: df2.any()  
[37]: arodes_int_num      True  
brzoza brodawkowata   True  
buk pospolity          True  
czereśnia pospolita   True  
dąb nieokreślony       True  
grab pospolity          True  
grusza pospolita       True  
jabłoń dzika           True  
jesion wyniosły         True  
kasztanowiec biały     True  
klon pospolity          True  
lipa drobnolistna      True  
modrzew europejski    True  
olsza czarna            True  
sosna zwyczajna         True  
wiąz pospolity          True  
wiśnia pospolita        True  
świerk pospolity        True  
topola                  True  
wierzba                 True  
gleba                   True  
funkcja_lasu             True  
uwilgotnienie            True  
siedlisko                True  
dtype: bool
```

Wszystkie kolumny zawierają chociaż jedną wartość prawdziwą.

Zapisuję tabelę końcową do pliku out.txt.

```
[37]: df2.to_csv('out.txt')
```

3. Sieci bayesowskie

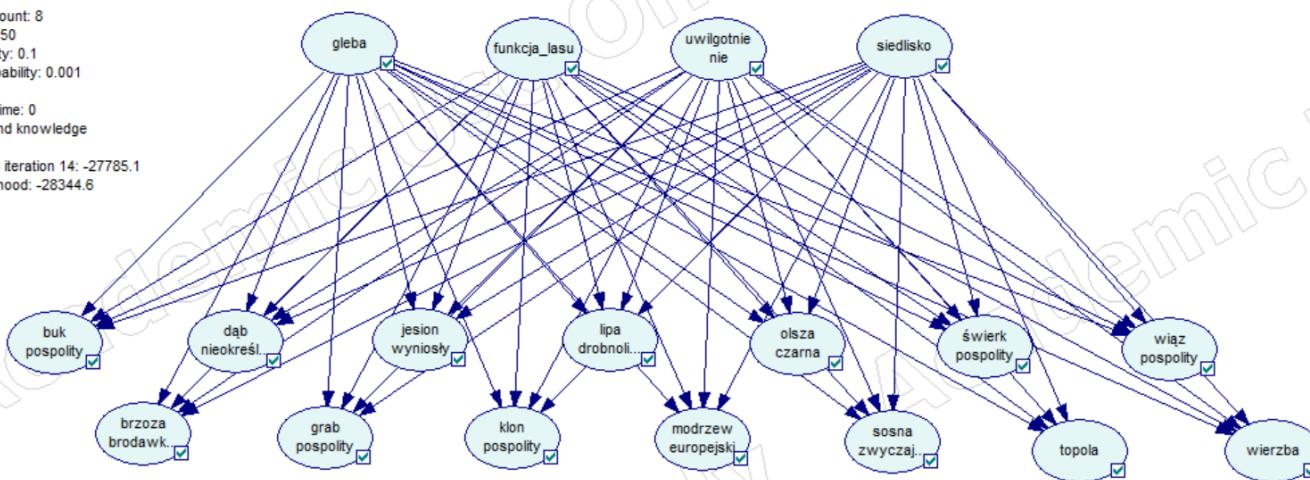
3.1 Sieć - graf dwudzielny

Tworzę sieć bayesowską z wszystkimi zmiennymi z uzyskanej tabeli (pomijając identyfikator powierzchni), gdzie połączenia będą prowadzić od zmiennych: rodzaj gleby, funkcja lasu, uwilgotnienie gleby i typ siedliska do wszystkich gatunków drzew.

Input file: out
Data rows: 5616
Elapsed time: 0.531s

Learning algorithm: Bayesian Search
Algorithm parameters:
Iterations: 20
Max parent count: 8
Sample size: 50
Link probability: 0.1
Prior link probability: 0.001
Seed: 0
Max search time: 0
No background knowledge

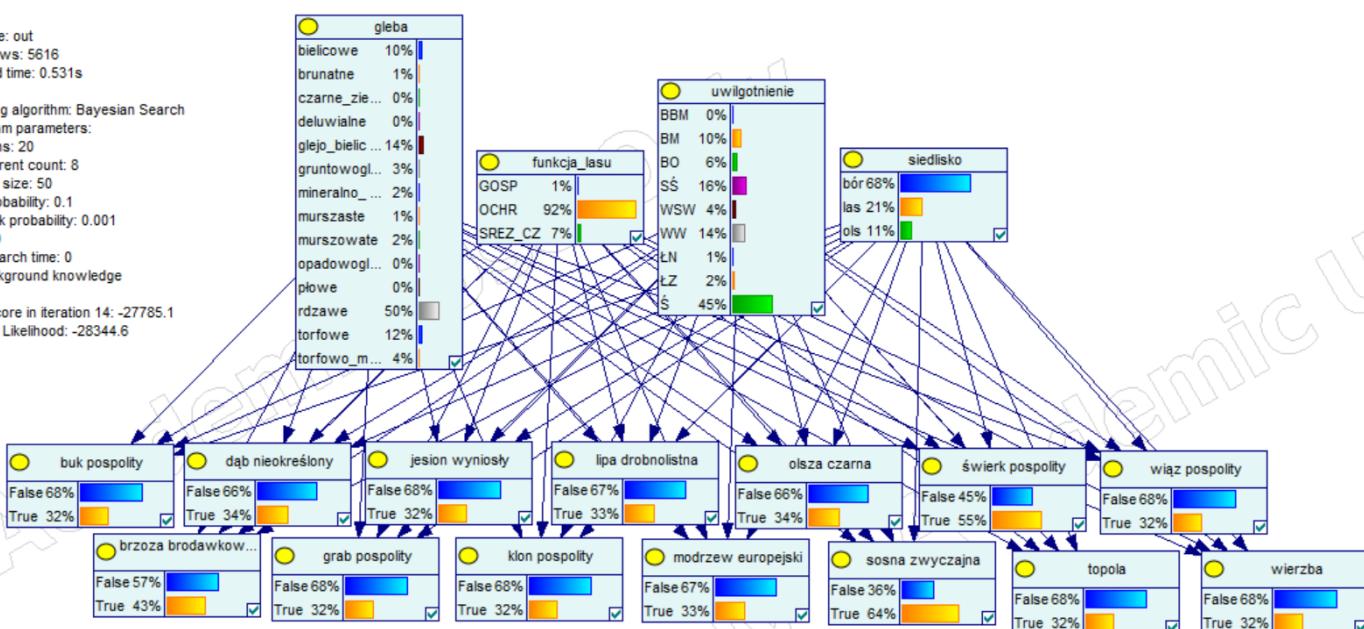
Best score in iteration 14: -27785.1
EM Log Likelihood: -28344.6



Input file: out
Data rows: 5616
Elapsed time: 0.531s

Learning algorithm: Bayesian Search
Algorithm parameters:
Iterations: 20
Max parent count: 8
Sample size: 50
Link probability: 0.1
Prior link probability: 0.001
Seed: 0
Max search time: 0
No background knowledge

Best score in iteration 14: -27785.1
EM Log Likelihood: -28344.6



Przy tworzeniu sieci, program pominął następujące gatunki drzew: czereśnia pospolita, grusza pospolita, jabłoń dzika, kasztanowiec biały, wiśnia pospolita.

3.1.1 Opis sieci (graf dwudzielny)

W sieci zostały drzewa liściaste: buk pospolity, brzoza brodawkowata, dąb, grab pospolity, jesion wyniosły, klon pospolity, lipa drobnolistna, olsza czarna, topola, wiąz pospolity, wierzba oraz drzewa iglaste: modrzew europejski, sosna zwyczajna, świerk pospolity. Wśród nich wyróżniamy 8 rodzin: bukowe (buk i dąb), brzozowe (brzoza, grab, olsza), oliwkowe (jesion), mydleńcowe (klon), ślazowe (lipa), wierbowe (topola, wierzba), wiązowe (wiąz) oraz sosnowe (modrzew, sosna, świerk).

Funkcje lasu, które są wyróżnione to: GOSP – lasy gospodarcze, OCHR – lasy ochronne oraz REZ_CZ – rezerwat częściowy.

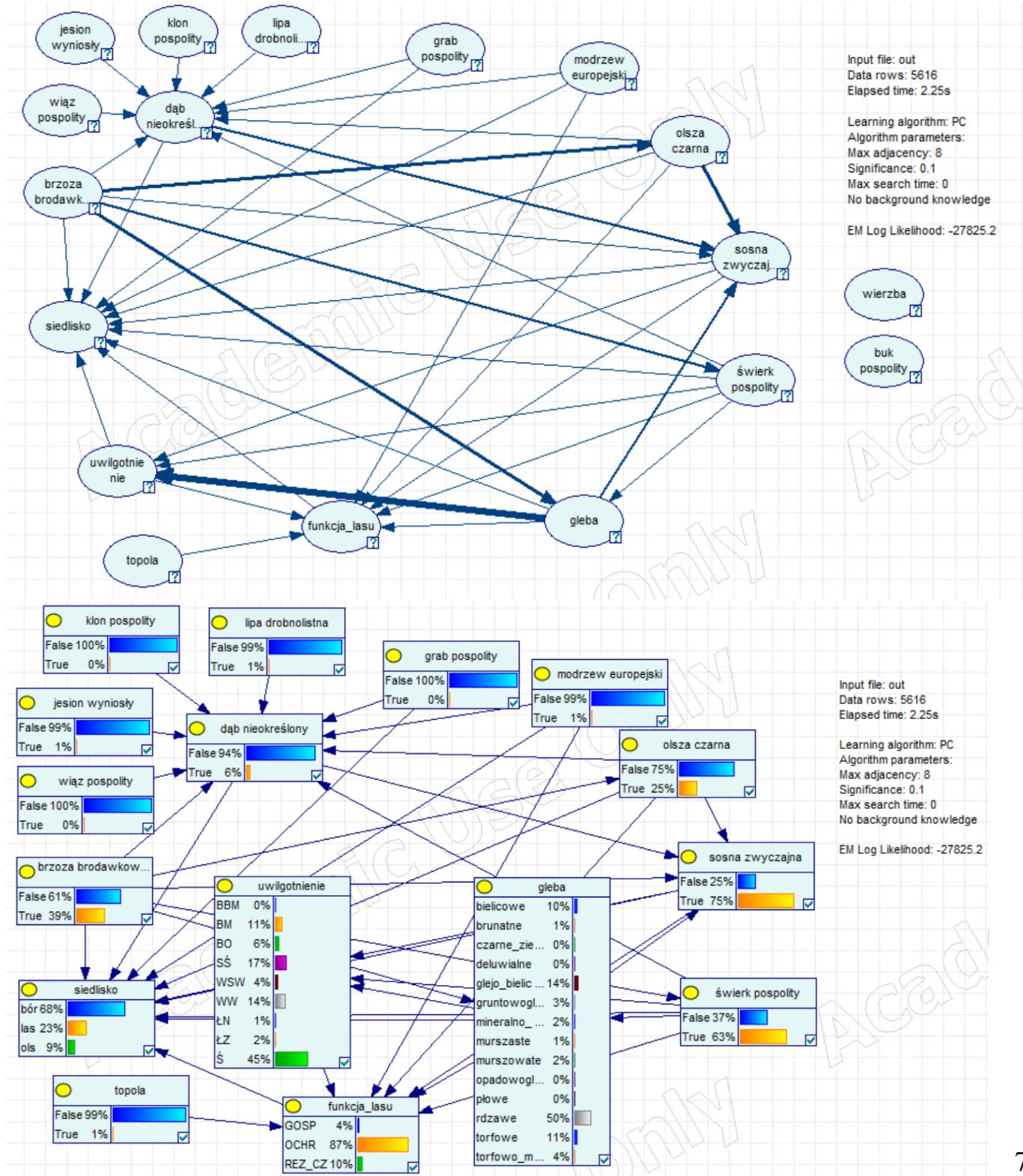
Z sieci widać, że na największej części powierzchni występują sosna zwyczajna i świerk pospolity. Największy udział w siedliskach ma bór, gleb najczęściej jest rdzawych, a funkcja lasu występująca najczęściej to las ochronny.

3.2 Wykrywanie przyczynowości

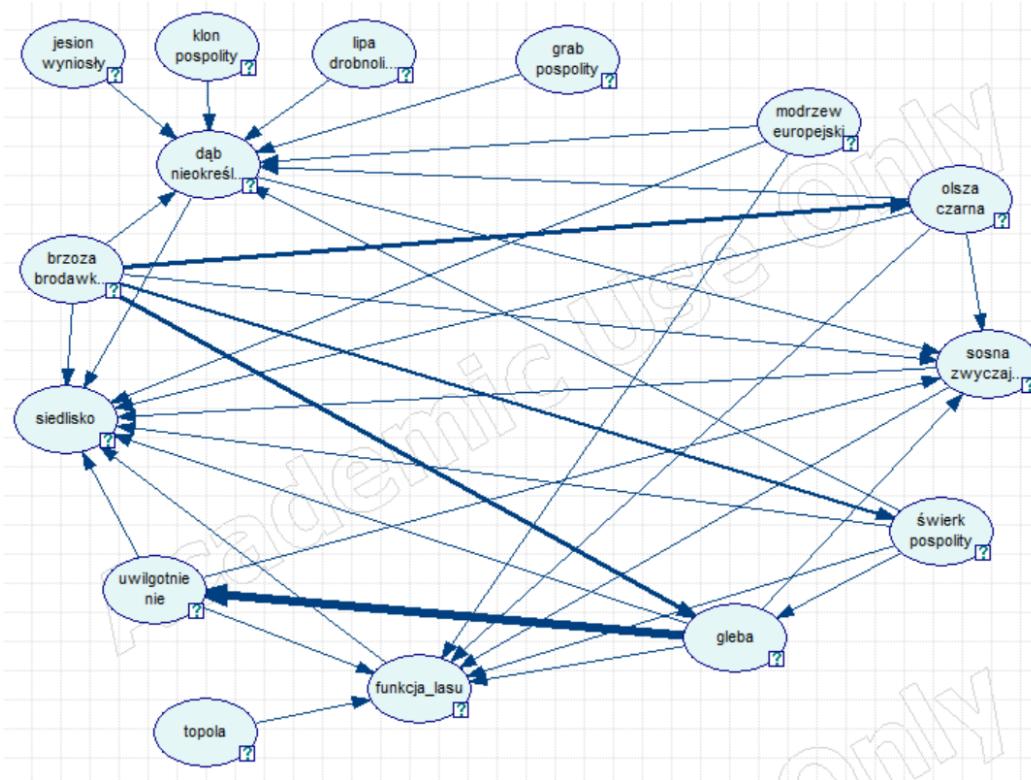
3.2.1 Sieci z różnymi poziomami istotności - algorytm PC

Wykonuję uczenie sieci Bayesowskiej z wykorzystaniem algorytmu PC z różnymi poziomami istotności testu niezależności zmennych.

1. $\alpha = 0,1$



2. $\alpha = 0,05$

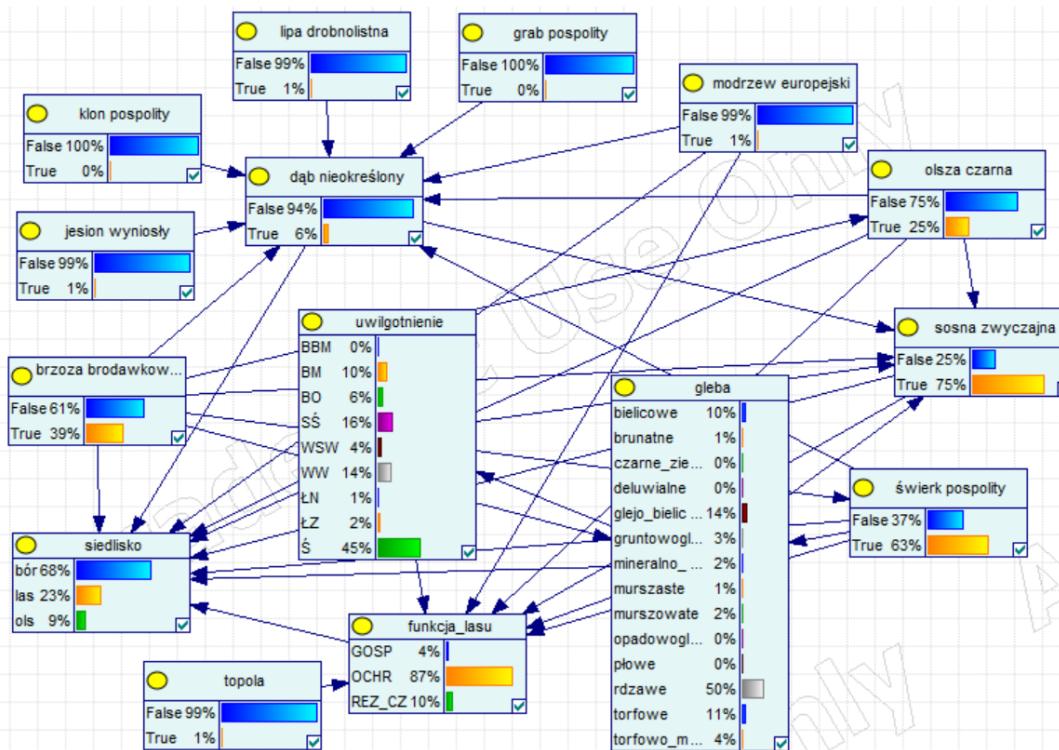


Input file: out
Data rows: 5616
Elapsed time: 1.891s

Learning algorithm: PC
Algorithm parameters:
Max adjacency: 8
Significance: 0.05
Max search time: 0
No background knowledge

EM Log Likelihood: -27865.8

wiąz pospolity
wierzba
buk pospolity

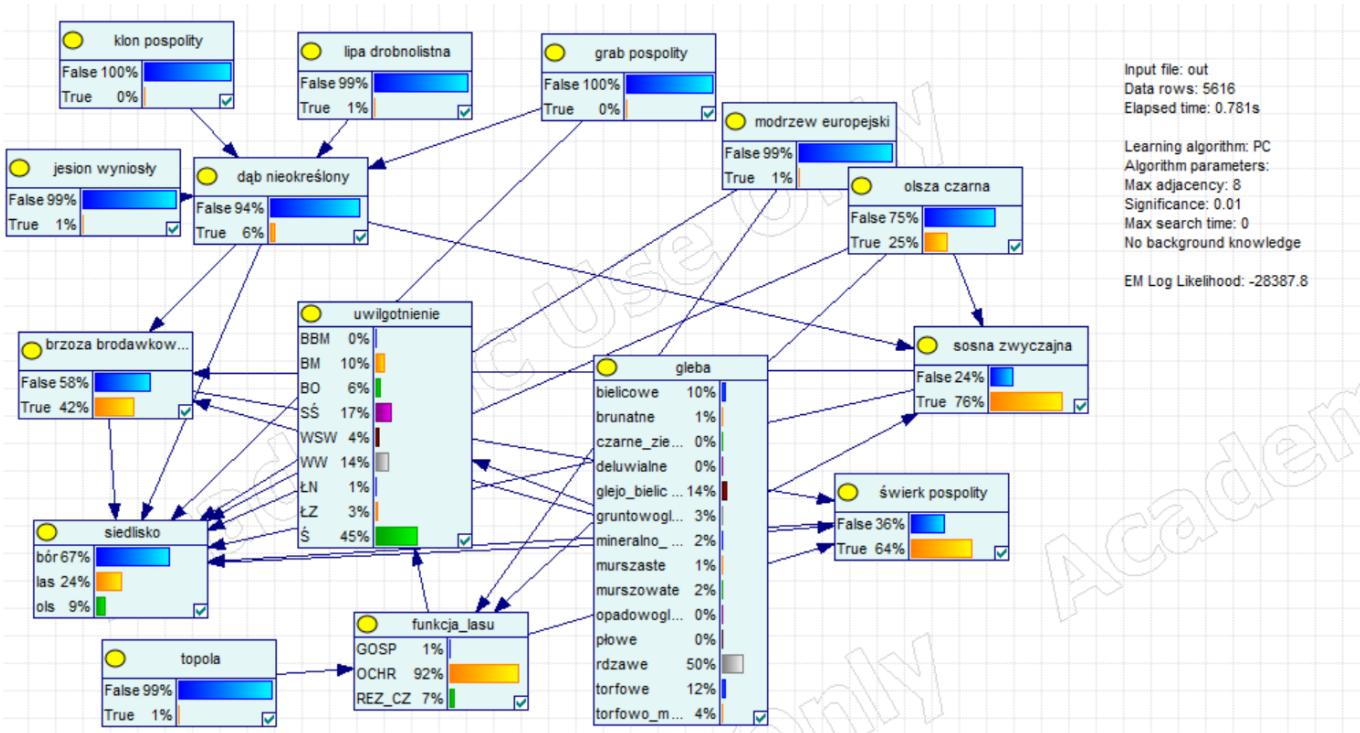
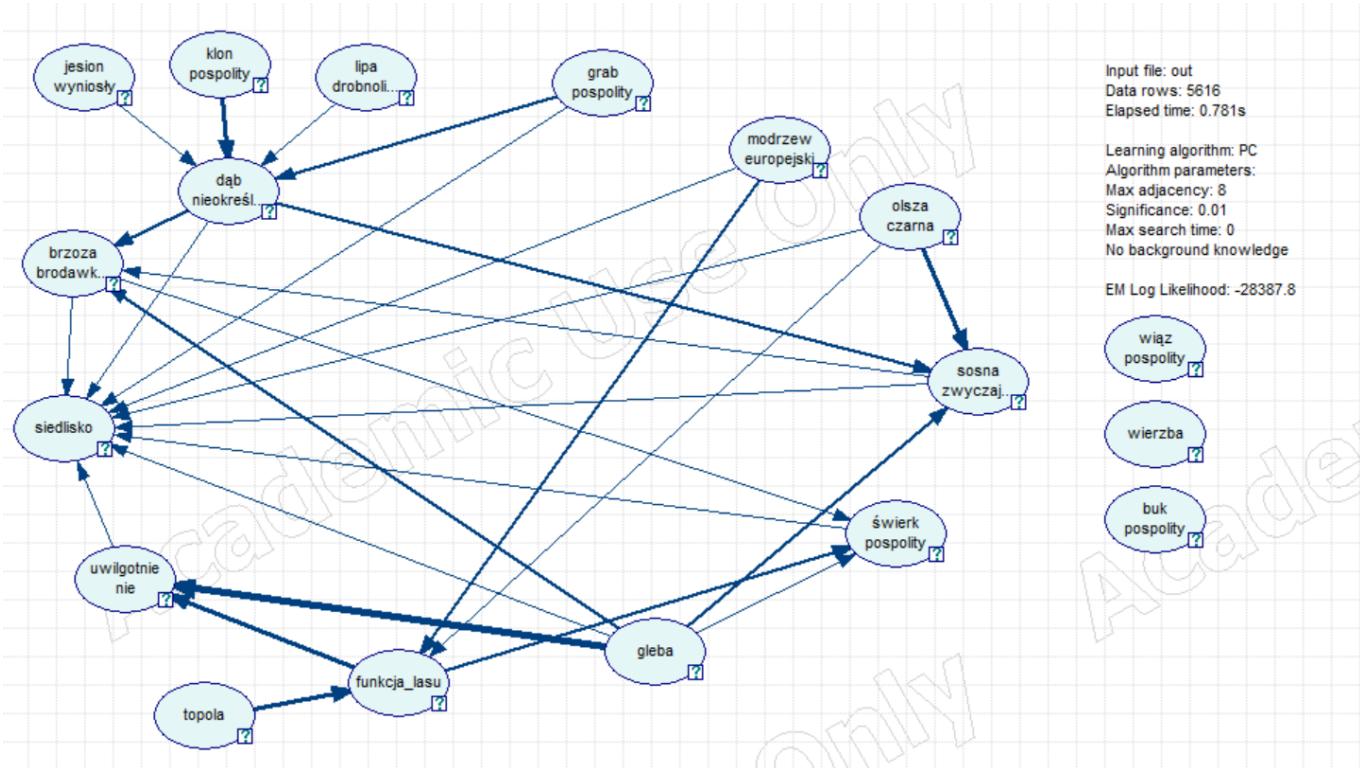


Input file: out
Data rows: 5616
Elapsed time: 1.891s

Learning algorithm: PC
Algorithm parameters:
Max adjacency: 8
Significance: 0.05
Max search time: 0
No background knowledge

EM Log Likelihood: -27865.8

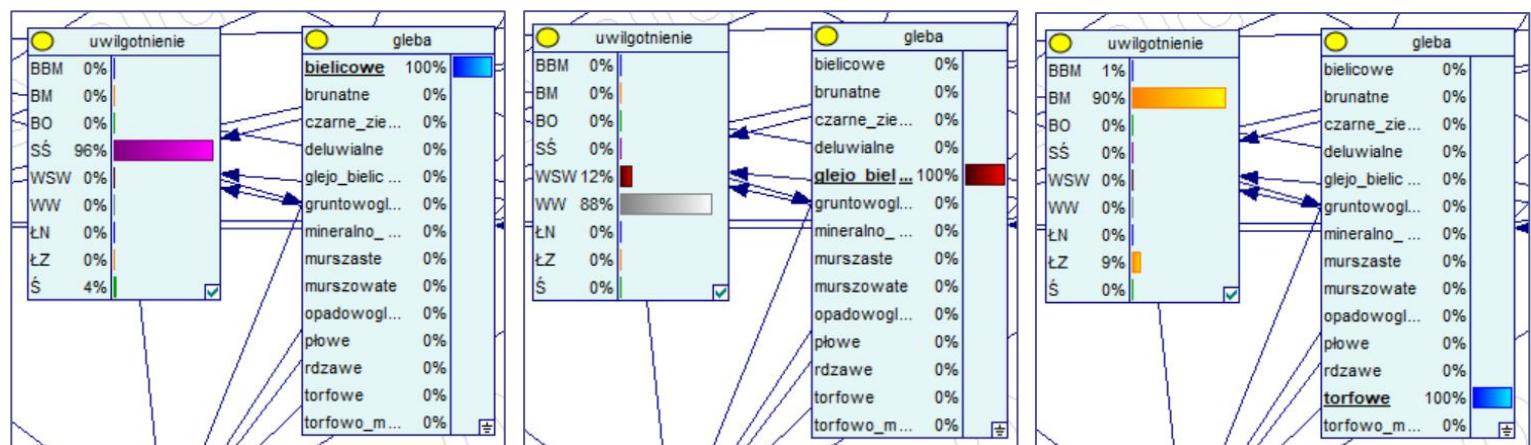
3. $\alpha = 0,01$



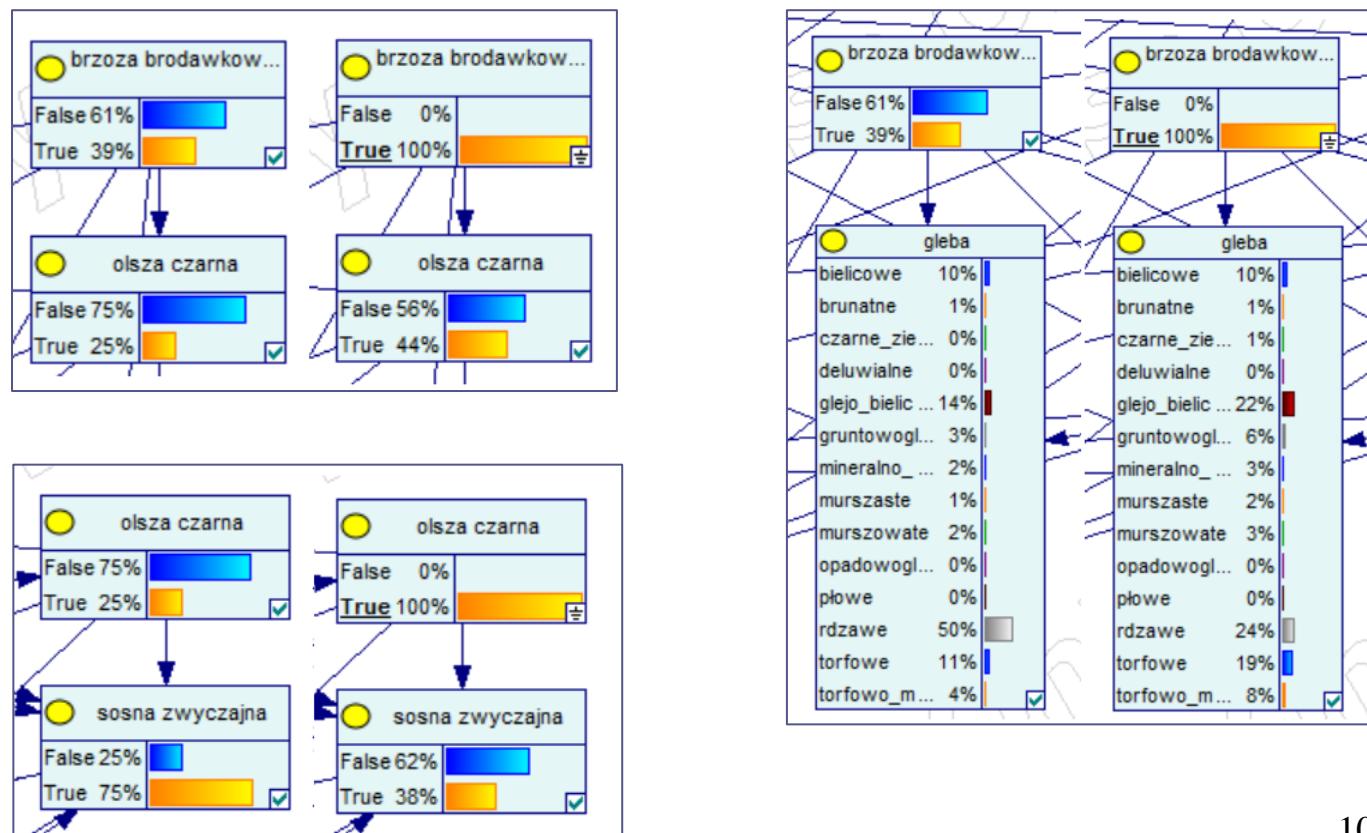
3.2.2 Opis sieci z różnymi poziomami istotności (PC)

W każdej z powyższych sieci widać silne oddziaływanie między rodzajem gleby, a wariantem uwilgotnienia. Dla każdej gleby można, z bardzo dużym prawdopodobieństwem, określić wariant uwilgotnienia.

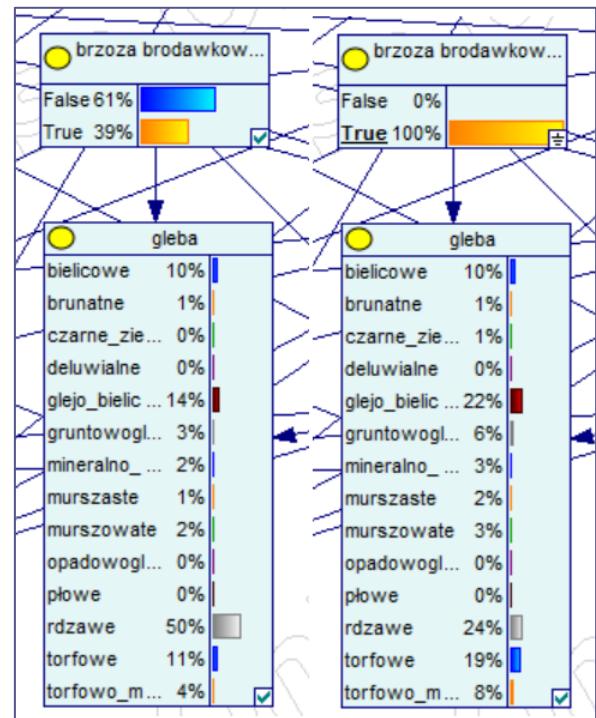
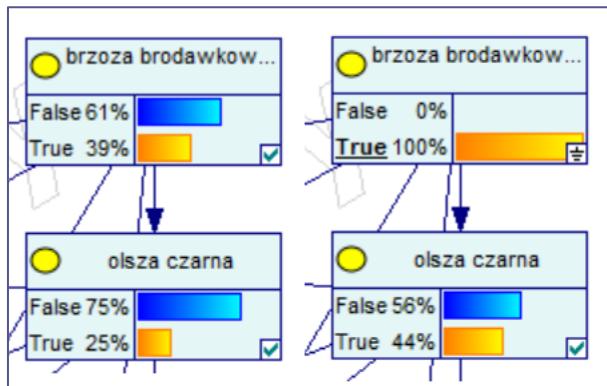
Przykład (sieć 1.)



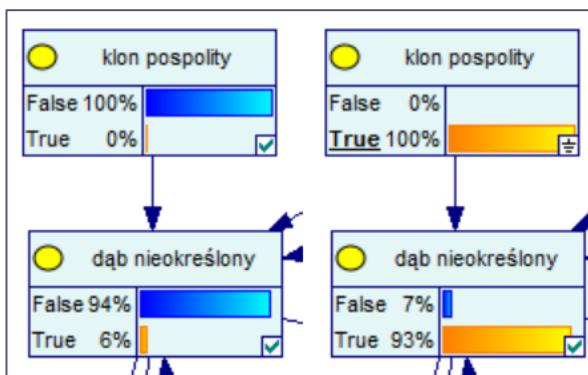
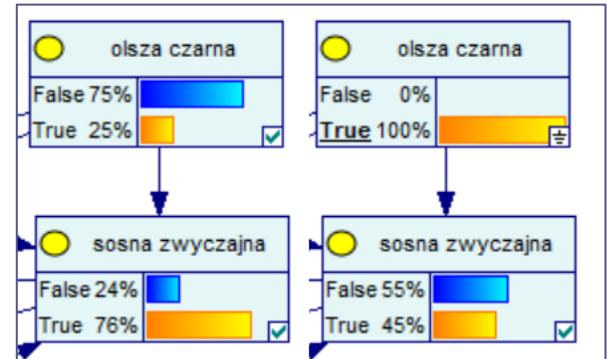
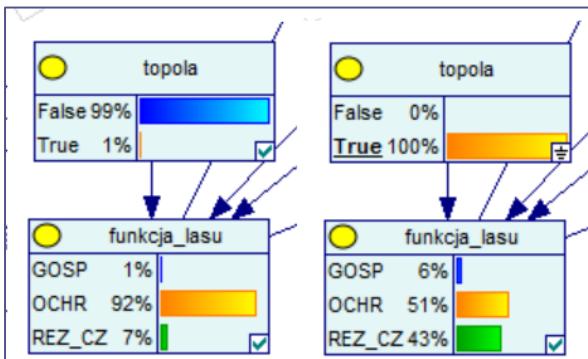
W sieci z poziomem istotności $\alpha = 0,1$ widocznie silne są również zależności między: występowaniem brzozy brodawkowej, a występowaniem olszy czarnej, występowaniem brzozy brodawkowej, a rodzajem gleby oraz występowaniem olszy czarnej, a występowaniem sosny zwyczajnej.

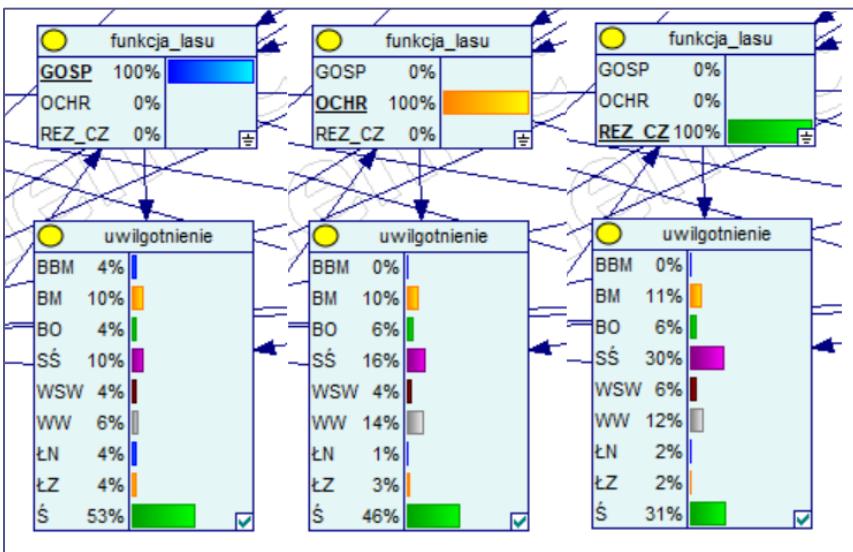


W sieci z poziomem istotności $\alpha = 0,05$ widoczne są, podobnie jak w sieci z poziomem istotności $\alpha = 0,1$, zależności między: występowaniem brzozy brodawkowej, a występowaniem olszy czarnej, występowaniem brzozy brodawkowej, a rodzajem gleby.



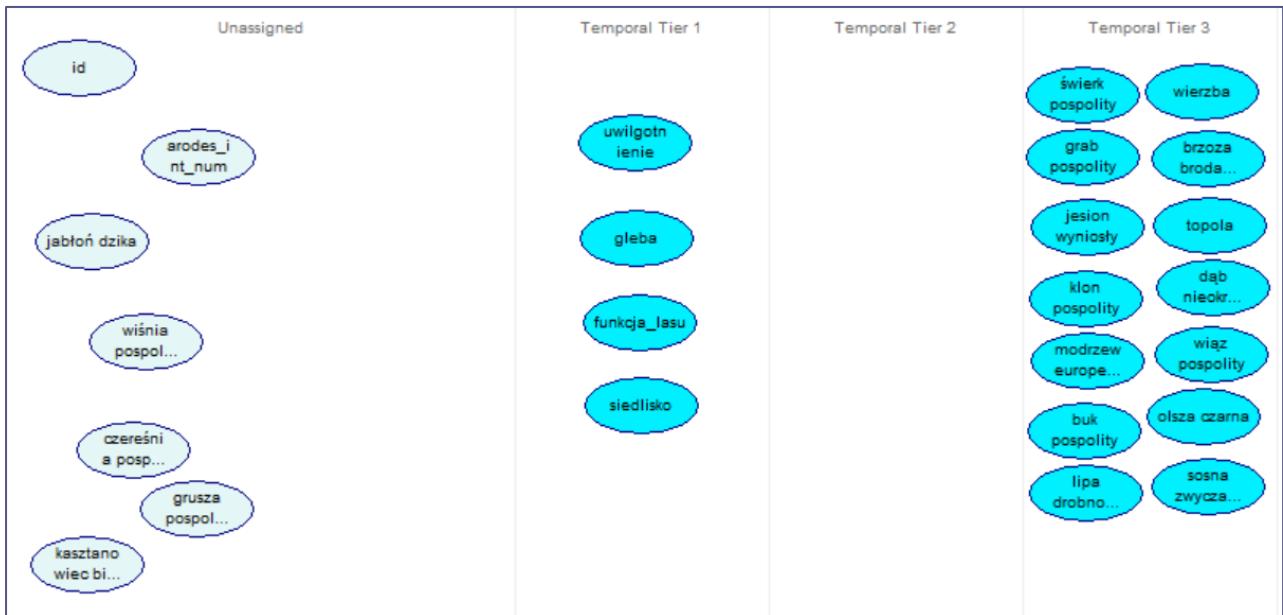
W sieci z poziomem istotności $\alpha = 0,01$ występują inne silne zależności: występowanie topoli i funkcja lasu, funkcja lasu i uwilgotnienie, występowanie olszy czarnej i sosny zwyczajnej, występowanie klonu pospolitego i dębu.





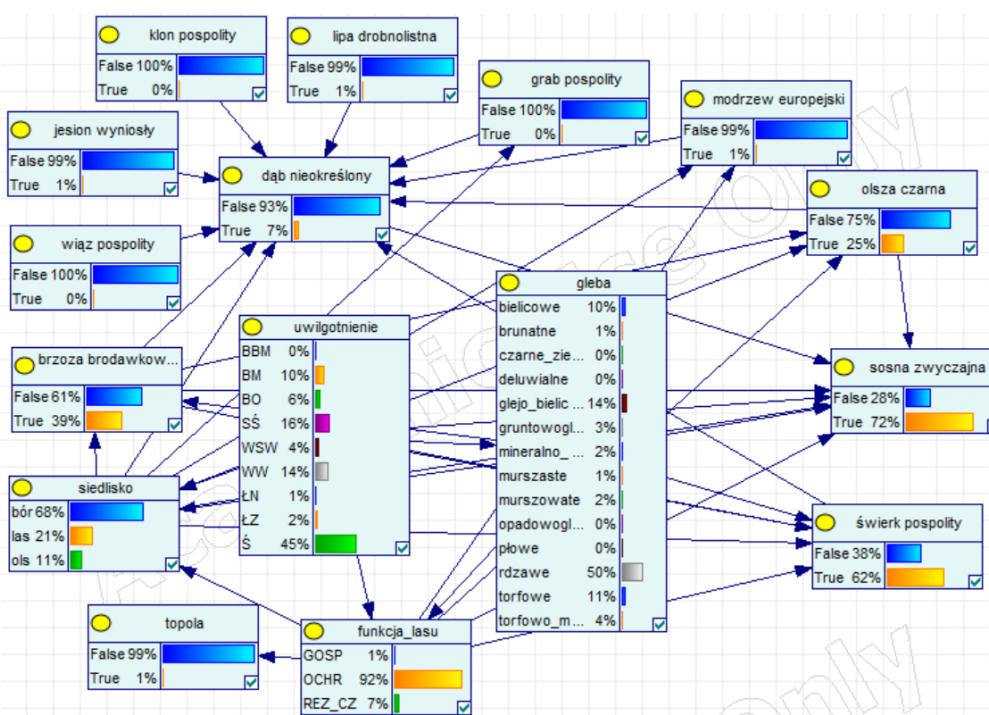
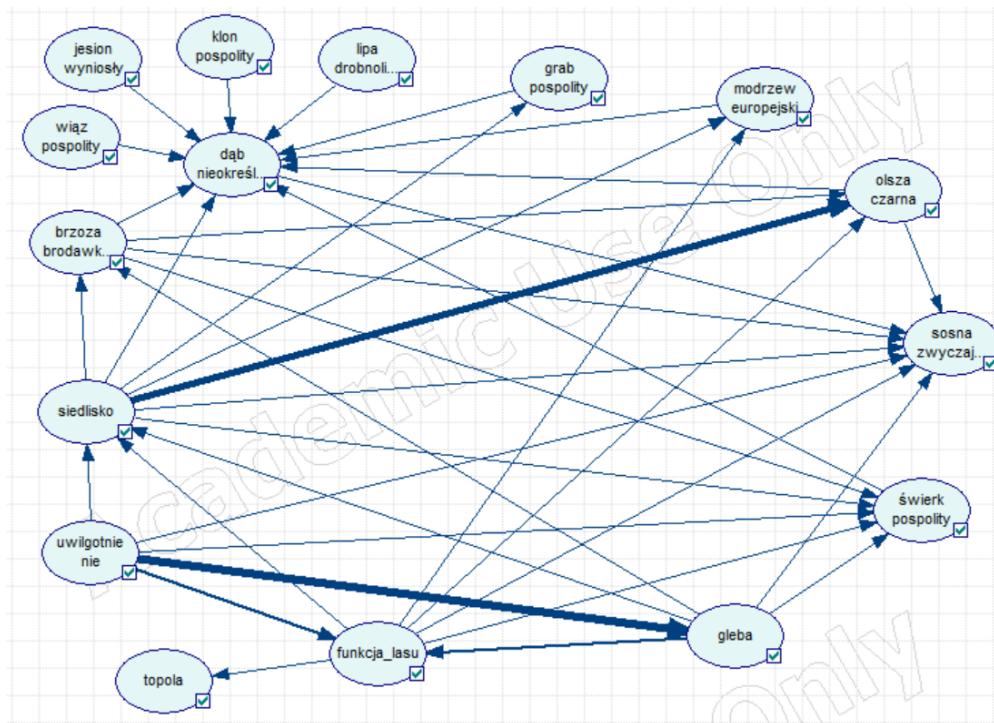
3.2.3 Background knowledge

Wykonuję poszczególne warianty ponownie, ustawiając wiedzę dodatkową (background knowledge), gdzie zmienne reprezentujące gatunki drzew są w "późniejszej" przestrzeni czasowej od pozostałych zmiennych (n-th tier).

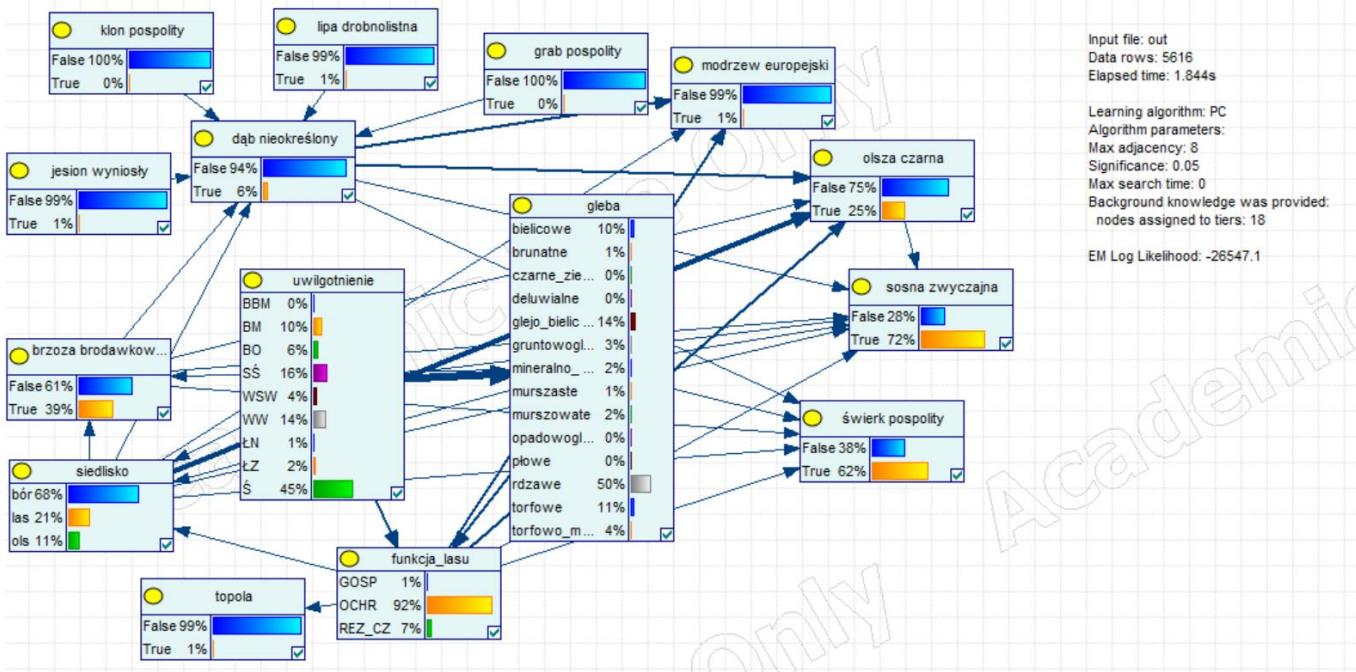
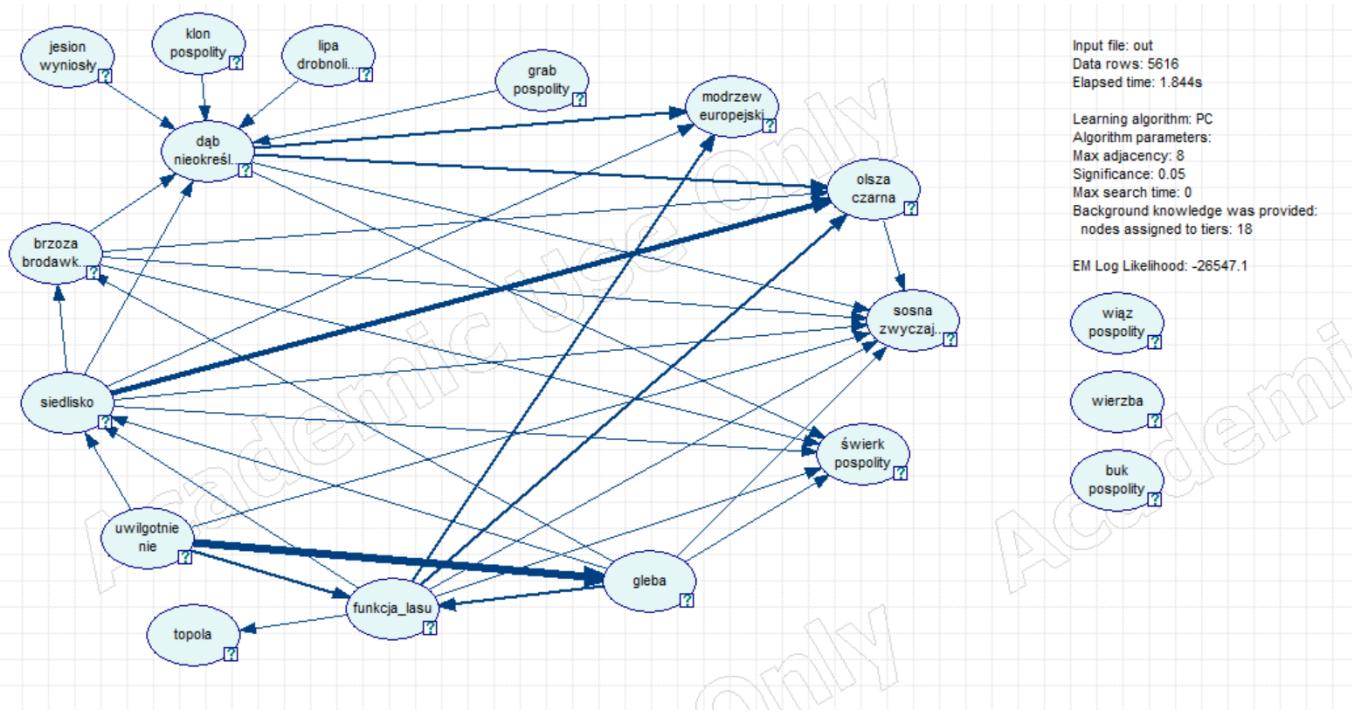


Ustawiam uwilgotnienie, glebę, funkcję lasu i siedlisko na tier 1, a znaczące gatunki drzew na tier 3.

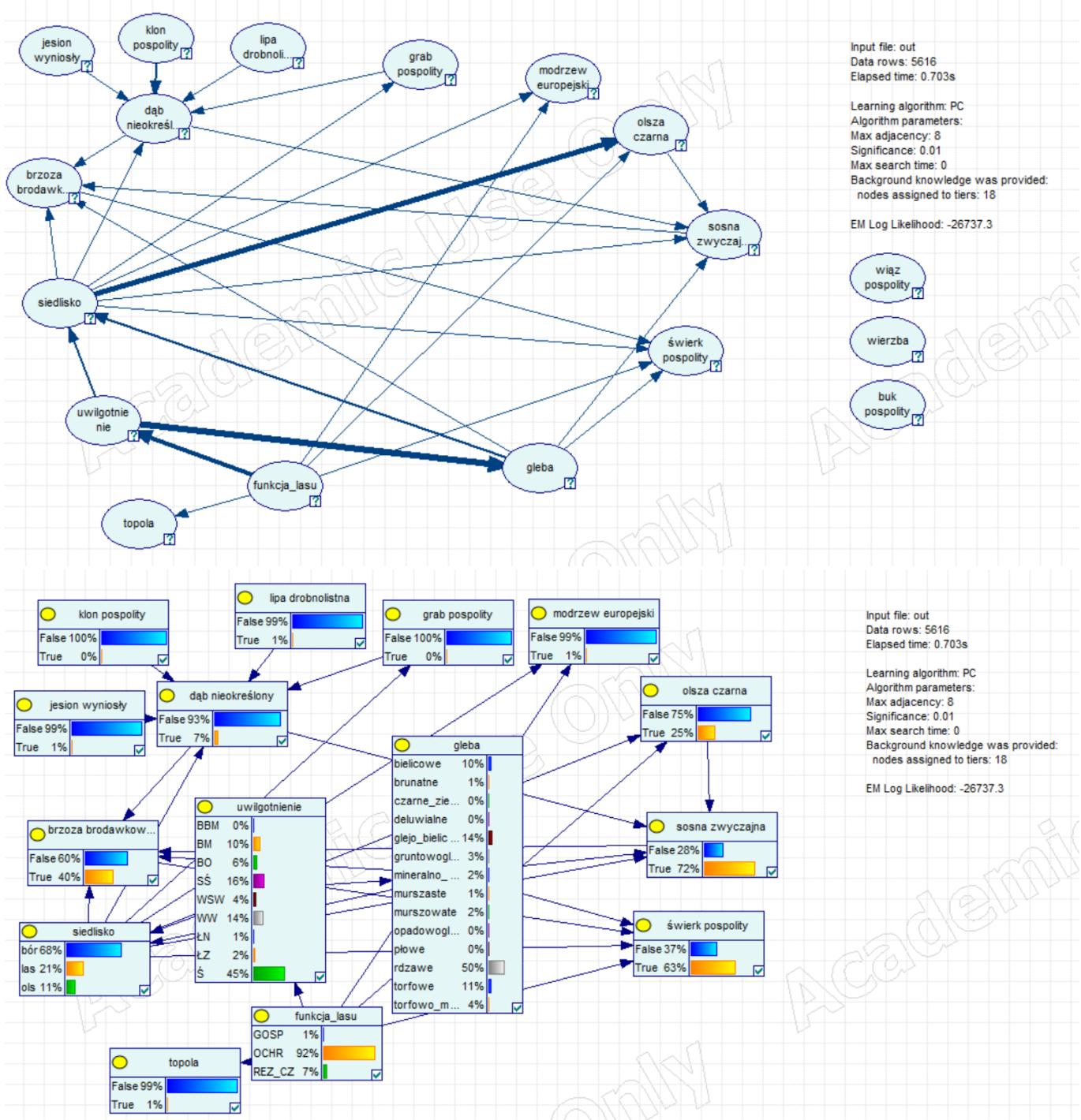
1. $\alpha = 0,1$



2. $\alpha = 0,05$



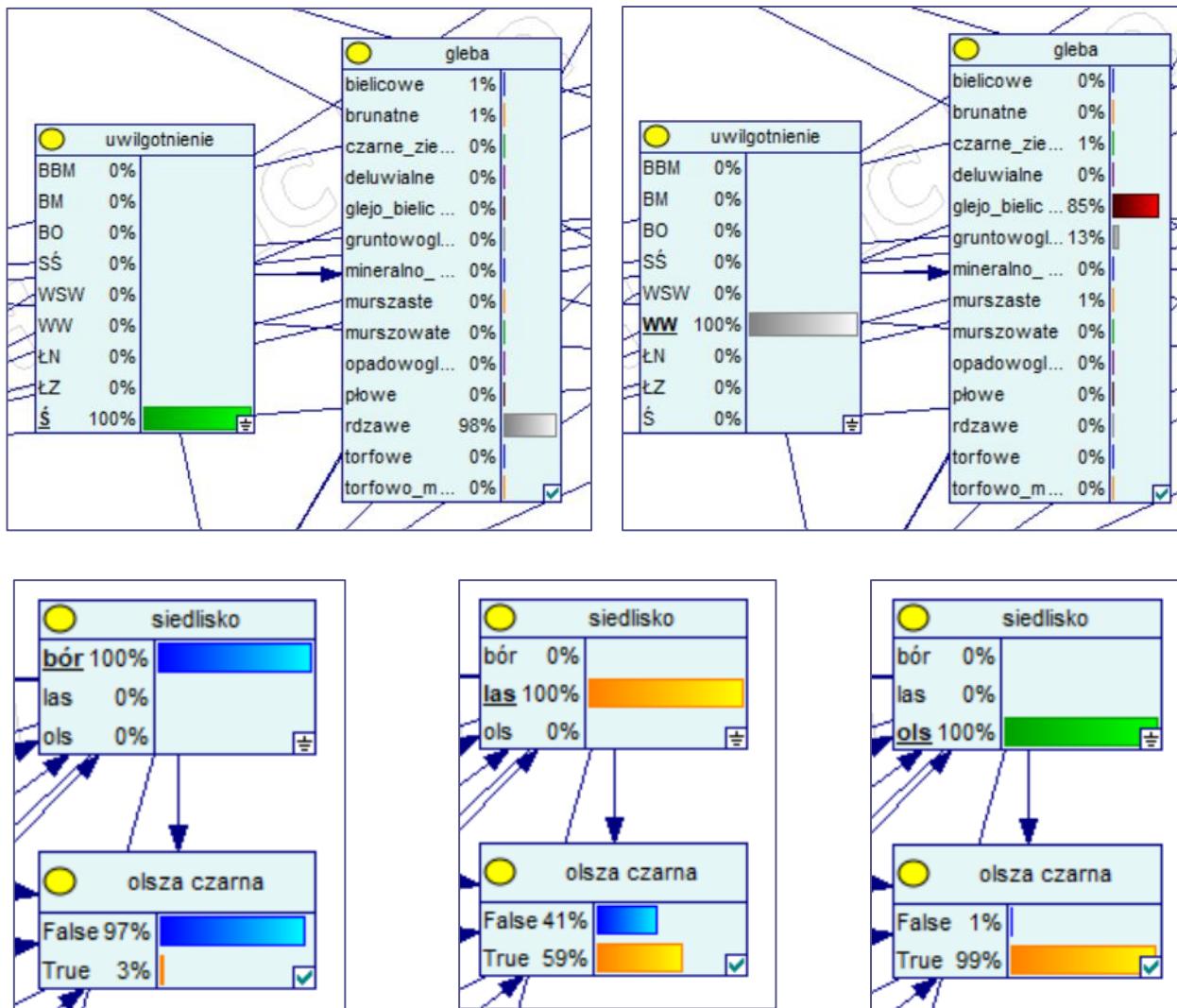
3. $\alpha = 0,01$



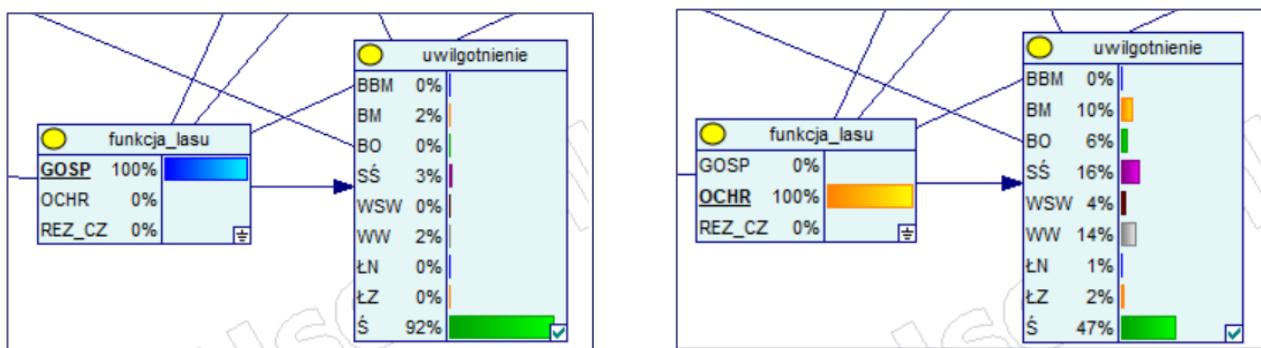
3.2.4 Opis sieci z background knowledge (PC)

W każdej z powyższych sieci jest bardzo silna zależność między wariantem uwilgotnienia, a rodzajem gleby oraz typem siedliska, a występowaniem olszy czarnej.

Przykład (sieć 2.)

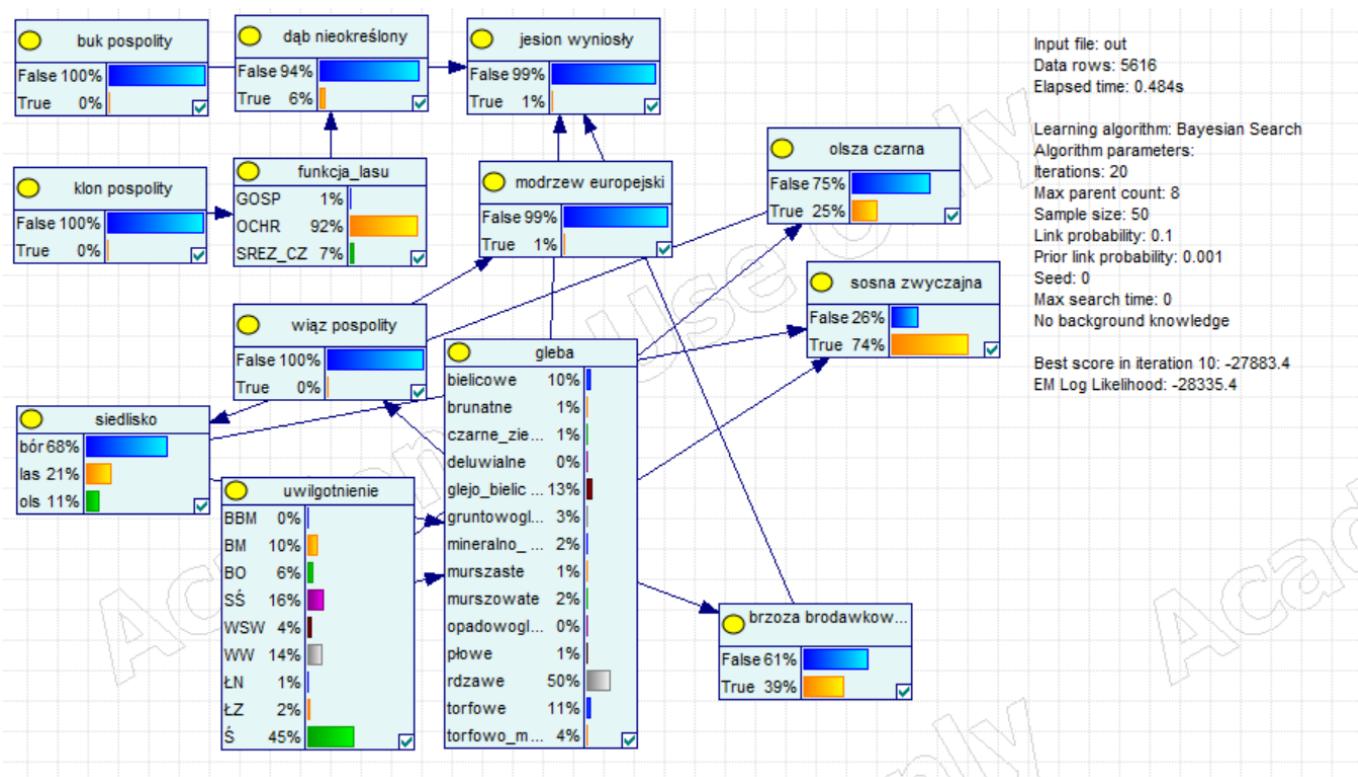
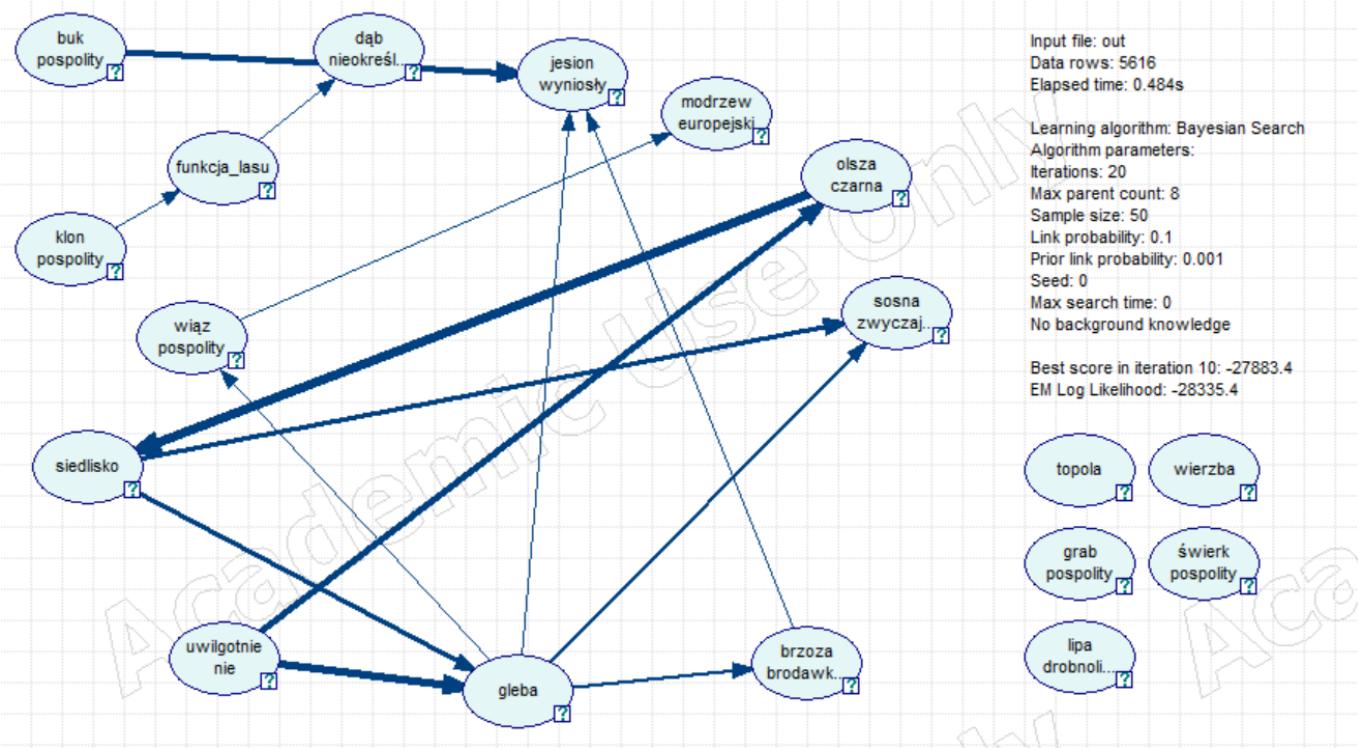


Inne zależności nie są tak silne. Można jedynie zwrócić uwagę na sieć z poziomem istotności $\alpha = 0,01$ gdzie dodatkowo widać zależność między funkcją lasu, a uwilgotnieniem.

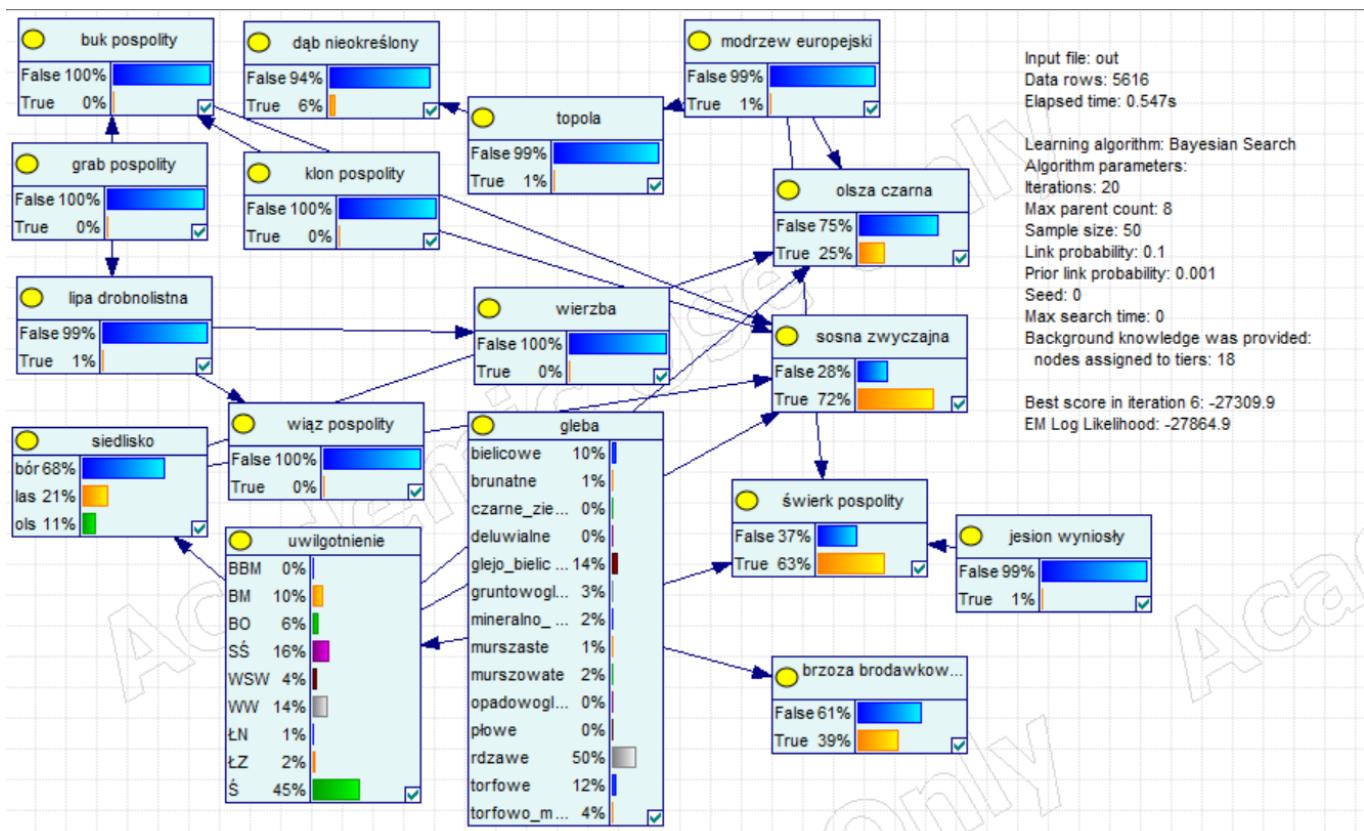
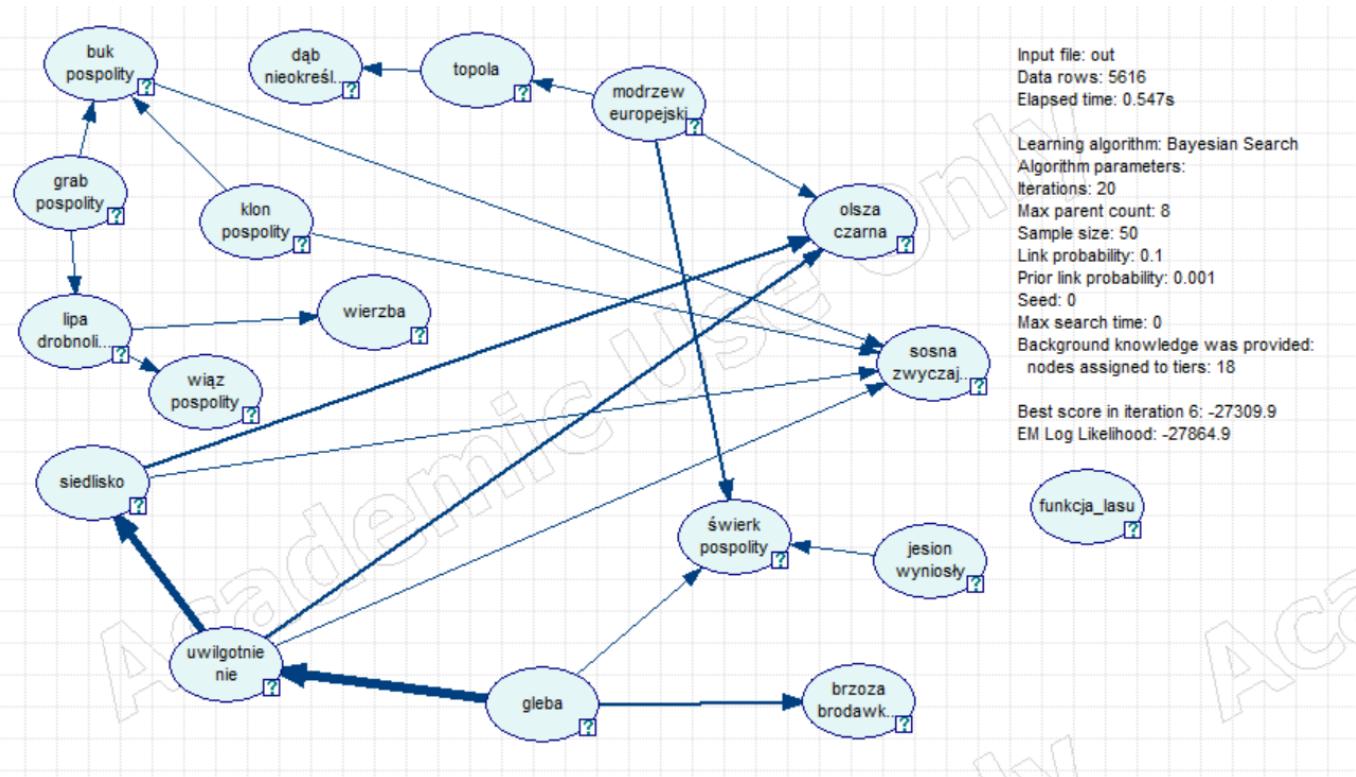


3.2.5 Algorytm Bayesian Search

1. Sieć bez background knowledge



2. Sieć z gatunkami drzew w późniejszej przestrzeni czasowej



3.2.6 Opis sieci (Bayesian Search)

Sieć 1. (bez background knowledge) cechuje się najmniejszą ilością węzłów. Zostało odrzucone 5 gatunków drzew, gdzie w sieciach generowanych algorytmem PC – maksymalnie 3. Widać, pojawiającą się już wcześniej, silną zależność między siedliskiem, a występowaniem olszy czarnej oraz między uwilgotnieniem, a rodzajem gleby. Można też zauważać nowe, niepojawiające się wcześniej, silne połączenia między: uwilgotnieniem, a występowaniem olszy czarnej, występowaniem buku, a występowaniem jesionu.

Sieć 2. (z drzewami w tier 3) ma najwięcej węzłów, jednak odrzucony węzeł to funkcja lasu, która może być istotną częścią sieci. Silne zależności występują między: uwilgotnieniem, a rodzajem gleby oraz uwilgotnieniem, a typem siedliska.

Obie sieci różnią się od siebie znacząco, w przeciwieństwie do sieci uczonych algorytmem PC, gdzie każda, niezależnie od poziomu istotności czy ustalonej wiedzy dodatkowej, miała podobne połączenia oraz liczbę węzłów.