

# SEMINARIO DEL 14/07

*Ruggieri Andrea*  
*Stranieri Francesco*  
*MAD Lab*



# OTTIMIZZAZIONE DEL CODICE

*Miglioramenti del codice*

*Parallelizzazione*

*Sviluppi futuri*

# OTTIMIZZAZIONE DEL CODICE

L'ottimizzazione del codice risulta essere fondamentale, dal momento che le reti possono essere estremamente complessi e il numero di missing value può essere alto.

Due tipi diversi di miglioramenti del codice sorgente sono stati fatti:

- Ottimizzazione del codice sulla base delle **best practices** di R
- **Parallelizzazione** del codice sorgente

N.B. Il lavoro di ottimizzazione è stato realizzato durante la fase di sperimentazione: il codice utilizzato per i dataset *Sports* e *Property* risulta essere non ottimizzato

# OTTIMIZZAZIONE DEL CODICE

L' ottimizzazione del codice sulla base delle **best practices** di R ha coinvolto diversi punti tra cui i più importanti:

- Uso di **datatable** e **accesso puntuale ai dati**, evitando cicli, ove possibile;
- Evitata la creazione di liste pesanti all'interno di cicli;
- Trasposizione dei dataframe originali (colonne sono rappresentate da variabili e righe da osservazioni);
- Rimozione di variabili ridondanti (pulizia del codice);
- Refactoring del codice, ove possibile.

# OTTIMIZZAZIONE DEL CODICE

Inoltre, per migliorare ulteriormente le performance si è deciso di parallelizzare l'esecuzione di EM HARD e di EM SOFT utilizzando la libreria Future di R <https://cran.r-project.org/web/packages/future/vignettes/future-1-overview.html>.

```
library("future")
plan(cluster, workers = 2, gc = TRUE)

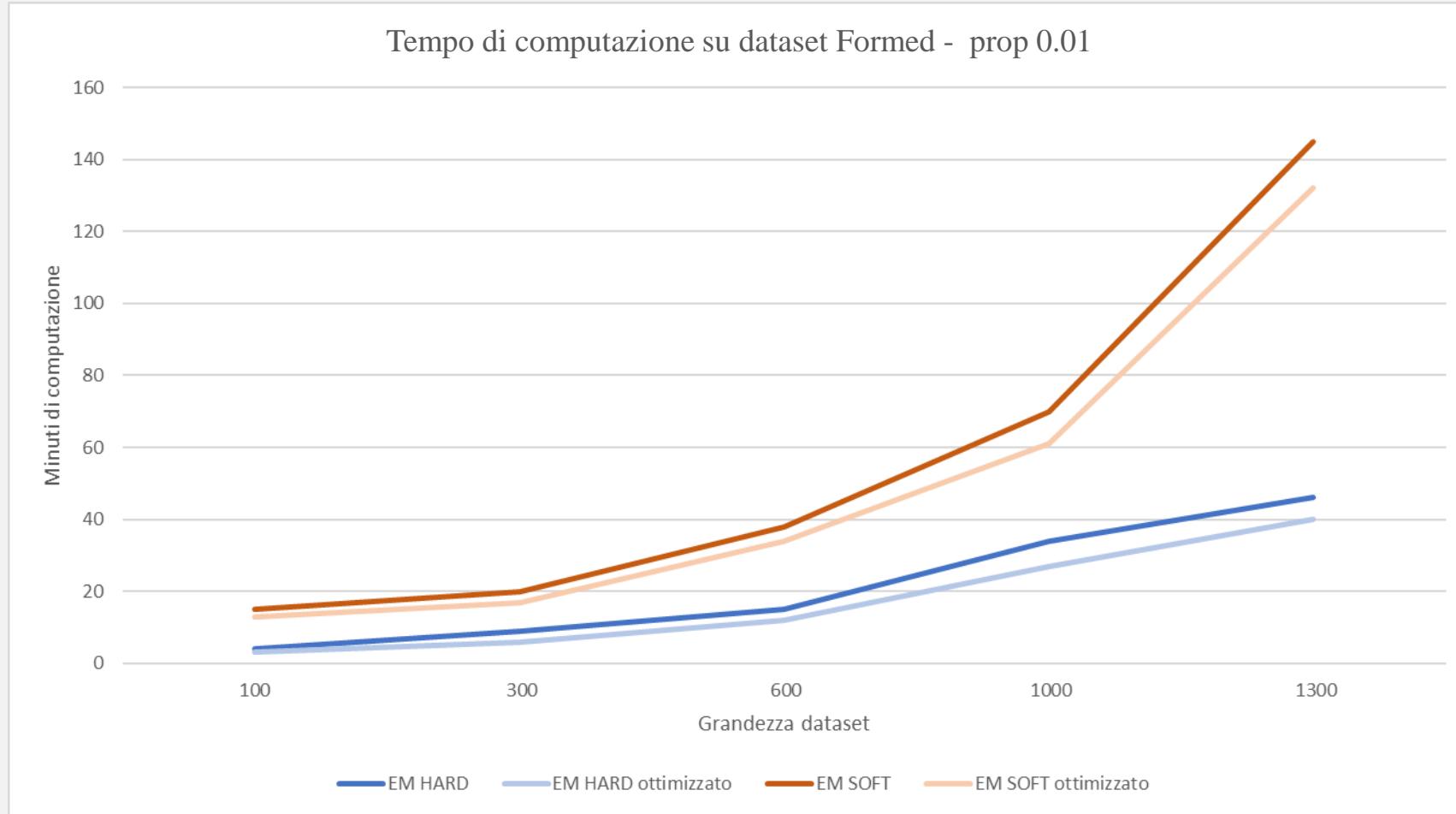
#Inizializzazione personalizzata custom
#print("INIZIO EM_HARD ASSIGNMENT")
results_soft %<-% em_soft(initial_data,structure,cpt,NUMBER_ITERACTION = 5)
Data_hard %<-% em_hard(initial_data,structure,cpt,NUMBER_ITERACTION = 3)

#print("INIZIO EM_SOFT ASSIGNMENT")
#print("INIZIO EM_SOFT_FORCED ASSIGNMENT")
Data_soft_forced <- results_soft$forced
Data_soft <- results_soft$soft

results_gt_hard <- compare_em_with_ground_truth_KL(net,Data_hard)
results_gt_soft <- compare_em_with_ground_truth_KL(net,Data_soft)
results_gt_soft_forced <- compare_em_with_ground_truth_KL(net, Data_soft_forced)

plan(sequential)
print("em end")
```

# OTTIMIZZAZIONE DEL CODICE

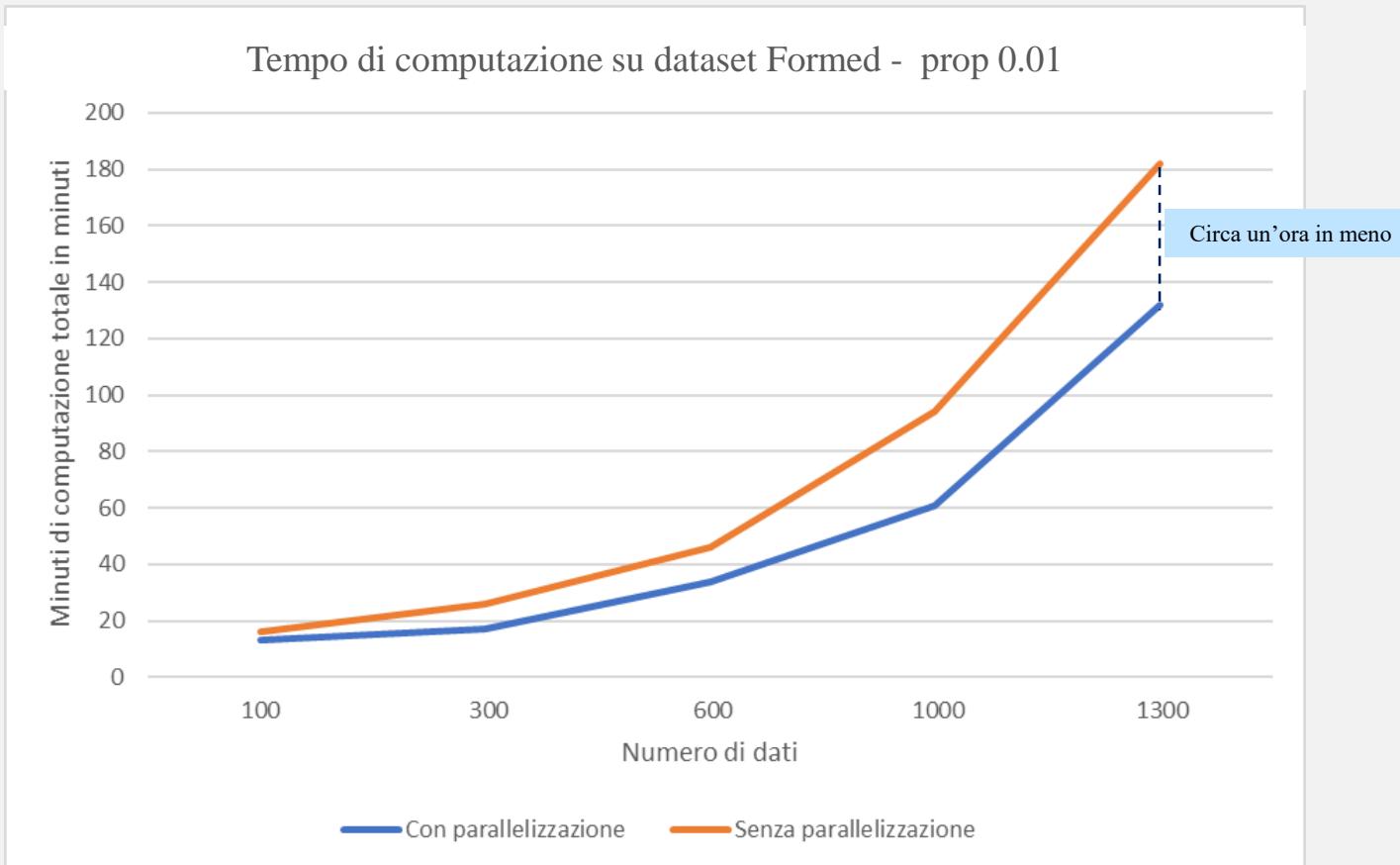


Tempo di computazione  
di una singola replica

Se numero di dati è pari  
a 1.300, il dataset è  
costituito da 123.200  
celle, tra cui il 5%  
missing (circa 2.464)

EM HARD forzato a 3  
iterazioni e EM SOFT a  
6 iterazioni.  
EM FORCED impiega  
un tempo analogo a EM  
HARD

# OTTIMIZZAZIONE DEL CODICE



Sommati i tempi di  
computazioni tra EM  
HARD e EM SOFT  
FORCED

# OTTIMIZZAZIONE DEL CODICE – SVILUPPI FUTURI

Nonostante queste ottimizzazioni, il codice risulta essere ancora ulteriormente migliorabile.

Possibili spunti:

- Modificare la tabella *table\_posterior\_prob* in modo tale che la ricerca non sia **quadratica** ma lineare.
- Ottimizzare l'inferenza esatta nello step di expectation in modo da non considerare tutte le variabili.
- Con opportune modifiche, modificare il metodo *update\_data\_hard* che in questo momento impiega un **tempo computazionale quadratico**.

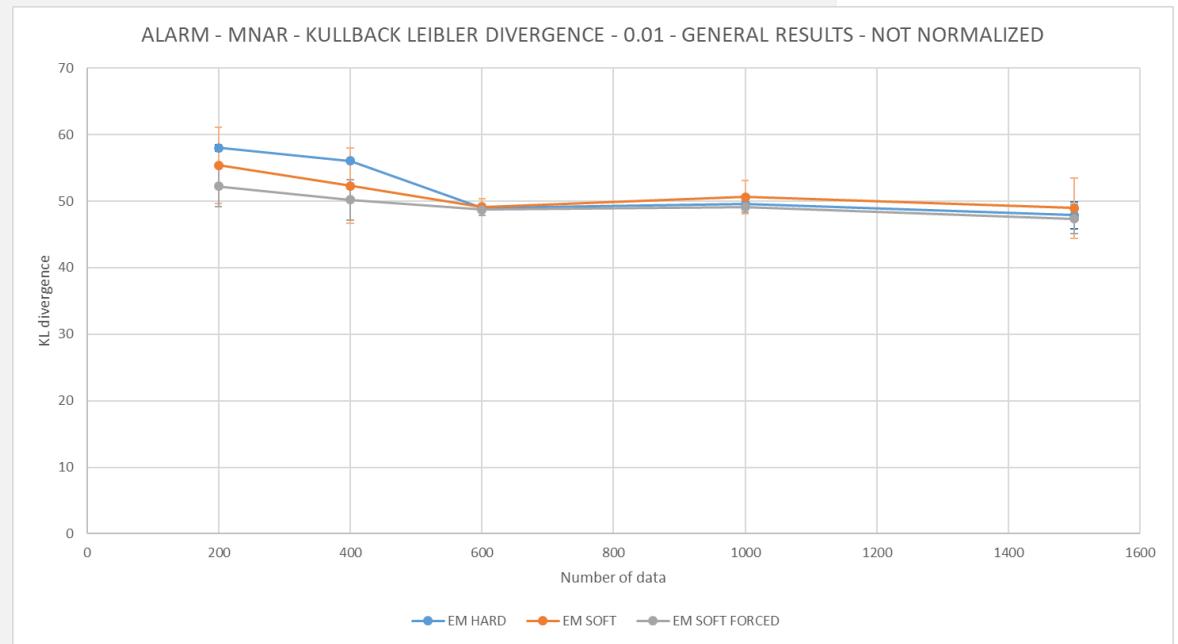
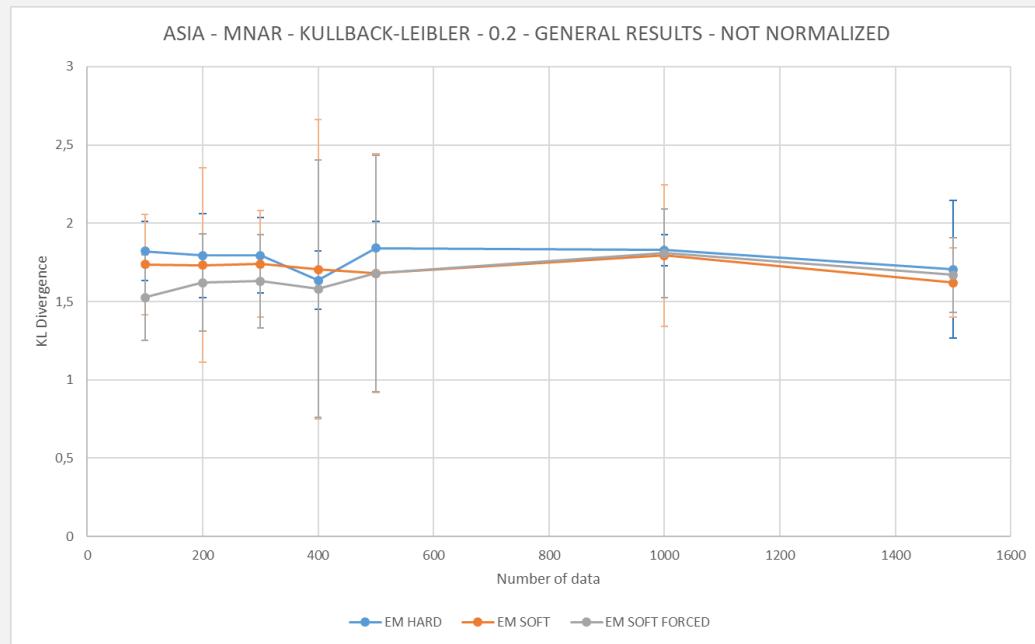
# INTRODUZIONE ALLE Sperimentazioni EFFETTUATE

*Approccio metodologico*

*Tipologia test effettuati*

# INTRODUZIONE AI TEST – RIASSUNTO

Nel precedente seminario si sono discussi i risultati ottenuti dalle reti ASIA e ALARM



Discutendo che non si riportavano differenze significative tra i tre tipi di algoritmi

# INTRODUZIONE AI TEST – APPROCCIO METODOLOGICO

Tuttavia, testare l'algoritmo EM su due dataset semplici come ASIA e ALARM, non è sufficiente al fine di giungere a conclusioni significative circa l'efficienza degli algoritmi EM HARD e EM SOFT

In questa presentazione, verranno dunque presentati i risultati ottenuti dalla computazione con dataset diversi:

- Rete di piccole dimensioni: **Sports**
- Rete di medie dimensioni: **Property**
- Rete di medie-grandi dimensioni: **ForMed**
- Rete di grandi dimensioni: **PathFinder**

# INTRODUZIONE AI TEST – APPROCCIO METODOLOGICO

## The Bayesys data and Bayesian Network repository

Anthony C. Constantinou<sup>1,2</sup>, Yang Liu<sup>1</sup>, Kiattikun Chobtham<sup>1</sup>, Zhigao Guo<sup>1</sup>, and Neville K. Kitson<sup>1</sup>

Version 1.02<sup>1</sup>

- i. [Bayesian Artificial Intelligence research lab](#), Risk and Information Management (RIM) research group, School of Electronic Engineering and Computer Science, Queen Mary University of London, London, UK, E1 4FZ.  
E-mail: [a.constantinou@qmul.ac.uk](mailto:a.constantinou@qmul.ac.uk)
- ii. The Alan Turing Institute, UK, British Library, London, UK, NW1 2DB.



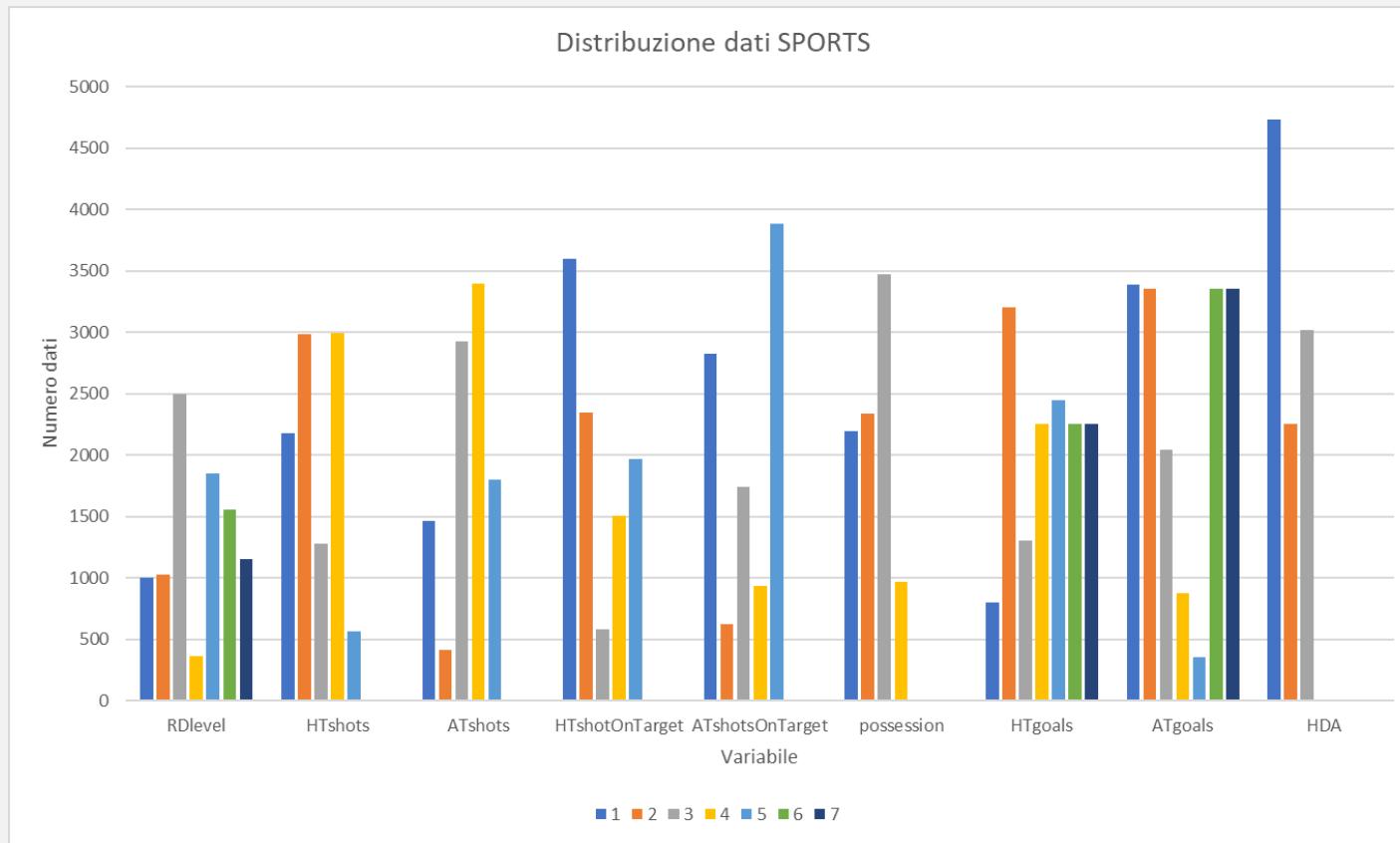
Tutti i dataset sono stati ottenuti e scaricati dal seguente link:

[http://constantinou.info/downloads/bayesys/bayesys\\_repository.pdf](http://constantinou.info/downloads/bayesys/bayesys_repository.pdf)

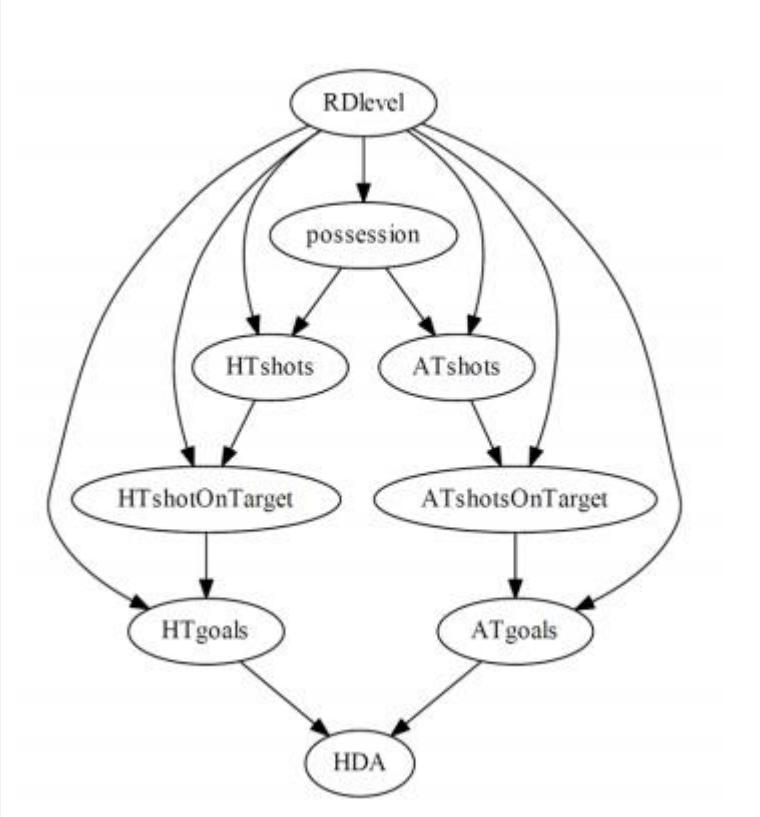
# INTRODUZIONE AI TEST – SPORTS

*Si tratta di una rete bayesiana di piccole dimensioni che combina le il livello di diverse squadre di calcio con varie statistiche sulle prestazioni della squadra per prevedere i risultati delle partite*

Il dataset si compone di 9 variabili, i cui i valori mostrano una tendenza ad essere bilanciati



# INTRODUZIONE AI TEST – SPORTS



La rete conta quindi:

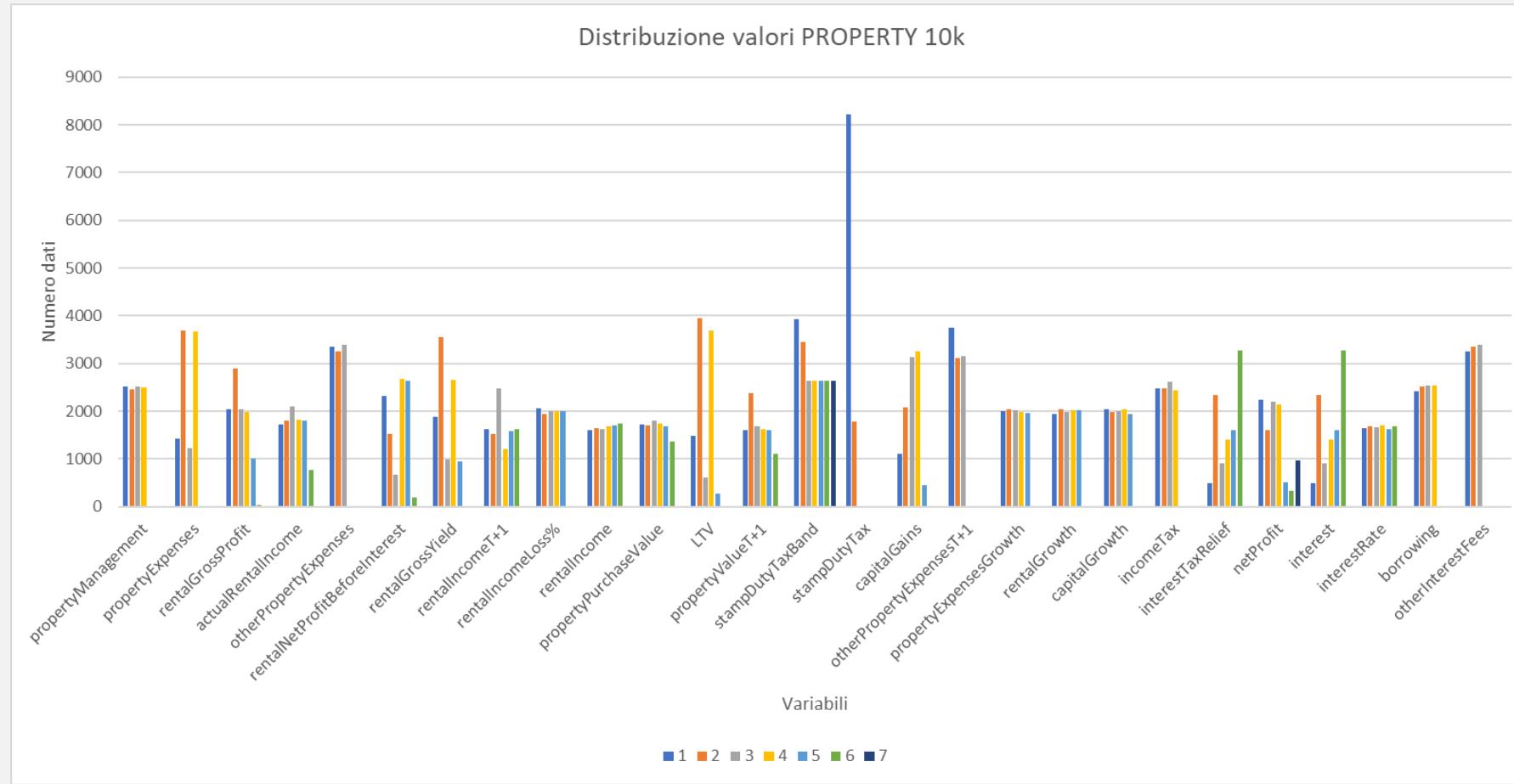
- 9 variabili
- 15 archi
- 1049 diversi parametri
- In-degree massimo 2
- Out-degree massimo: 7
- Numero di foglie: 1
- Numero di radici: 1

**HDA** viene considerato l'attributo di classe, contiene 3 possibili valori (vittoria squadra di casa, pareggio o vittoria squadra ospite)

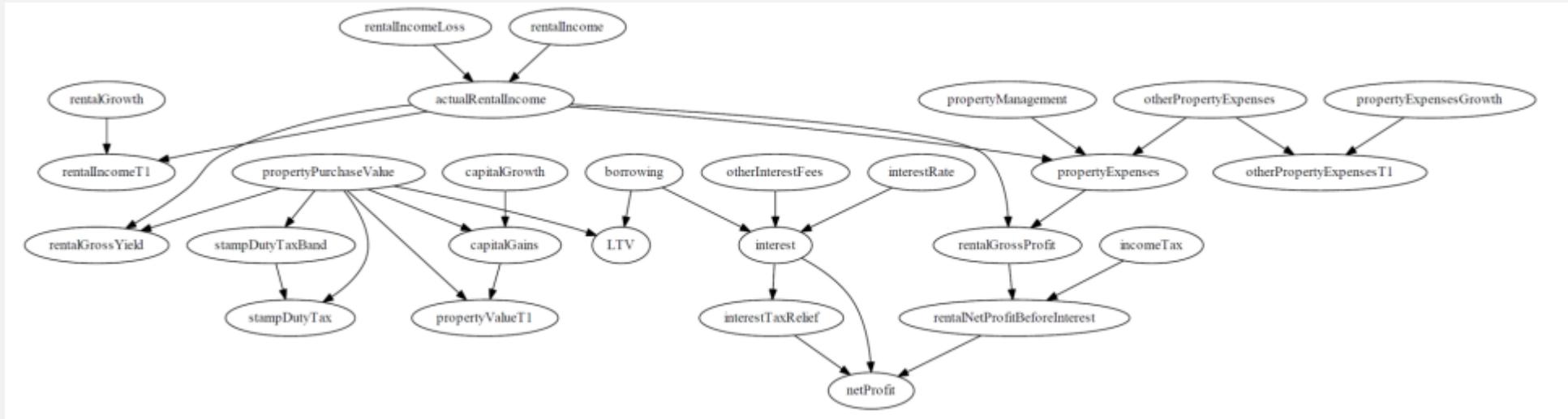
# INTRODUZIONE AI TEST – PROPERTY

*Si tratta di una rete bayesiana di medie dimensioni che valuta le decisioni di investimento nel mercato immobiliare britannico*

Il dataset si compone di 27 variabili, i cui i valori mostrano una tendenza ad essere bilanciati



# INTRODUZIONE AI TEST – PROPERTY



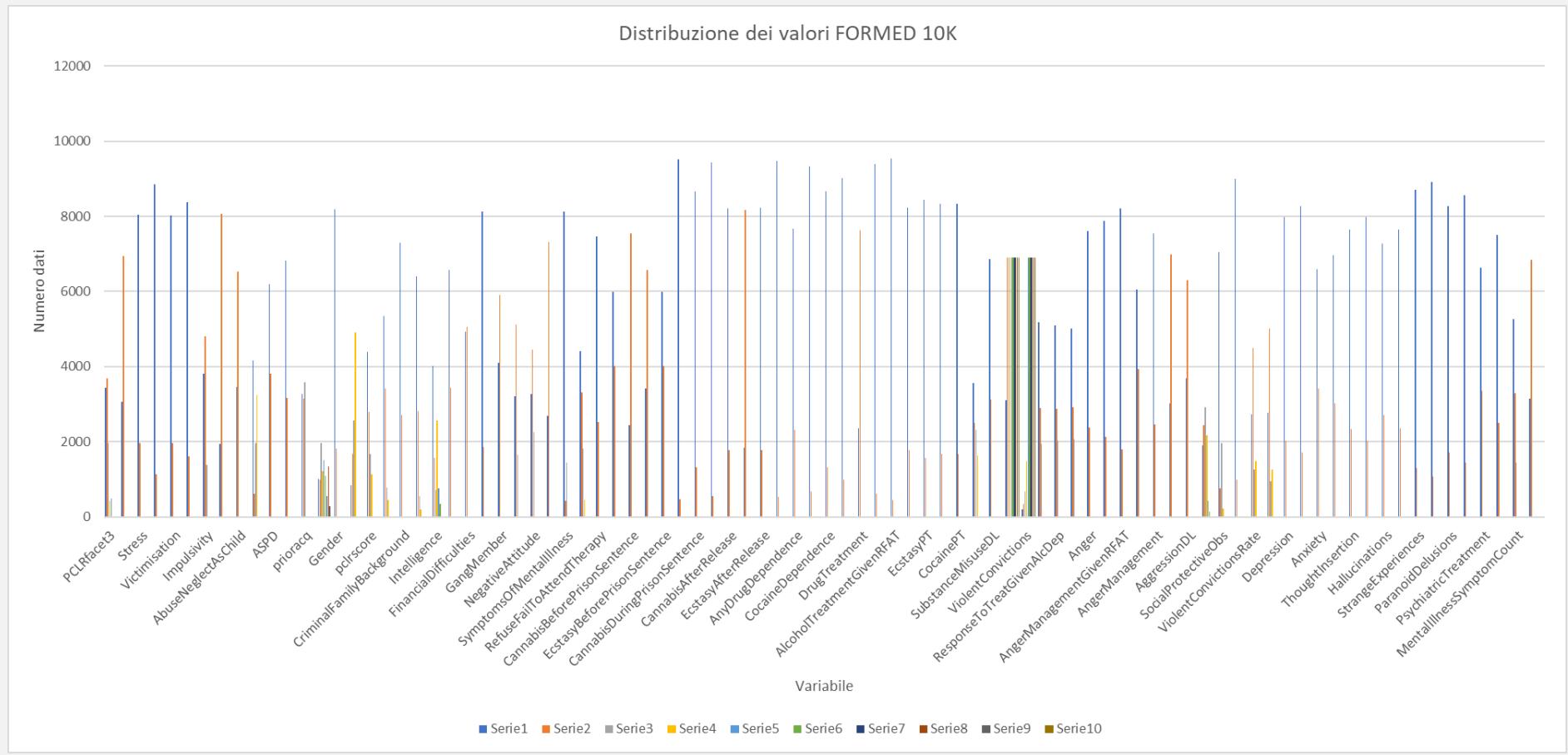
La rete conta quindi:

- 27 variabili
- 31 archi
- 3056 diversi parametri
- In-degree massimo 3
- Out-degree massimo: 6
- Numero di foglie: 7
- Numero di radici: 11

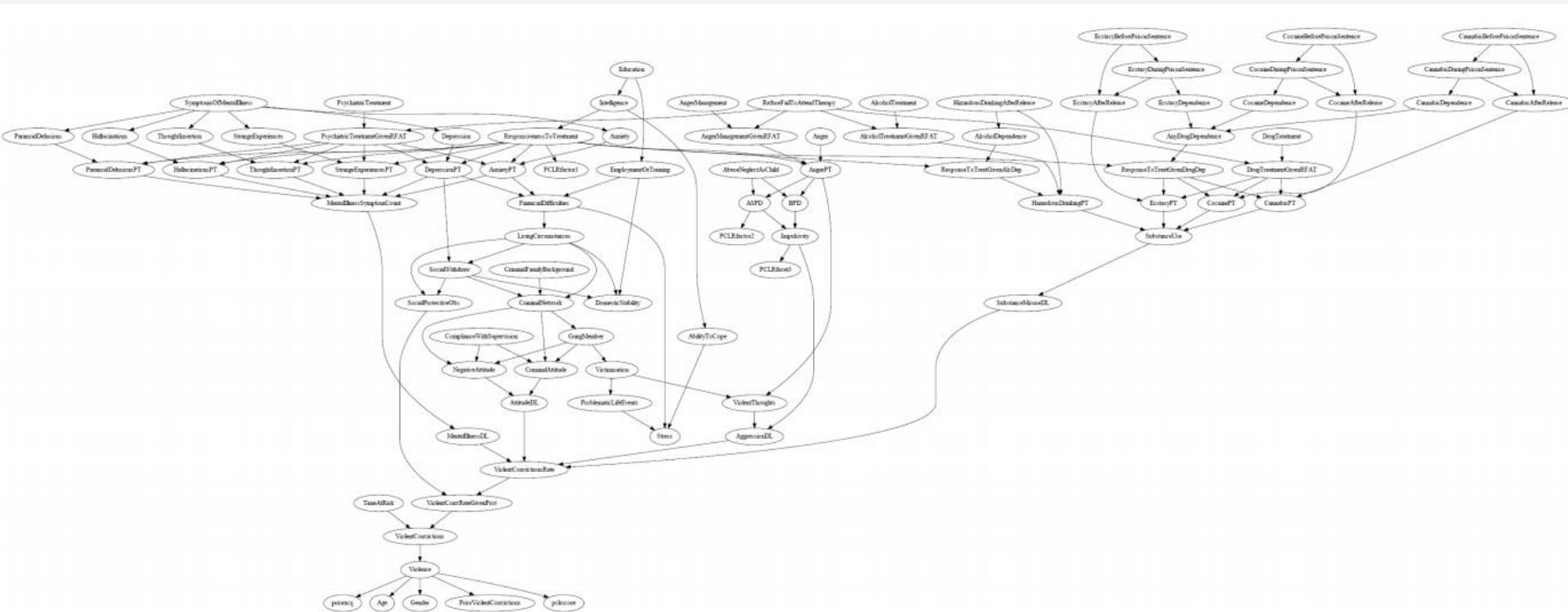
# INTRODUZIONE AI TEST – FORMED

*Una rete bayesiana di medie-alte dimensioni che cattura il rischio di recidiva violenta dei prigionieri malati di mente, insieme a una serie di possibili interventi per la gestione di questo rischio*

Il dataset si compone di 88 variabili, la maggior parte delle variabili sono binarie, è presente un lieve sbilanciamento dei valori



# INTRODUZIONE AI TEST – FORMED



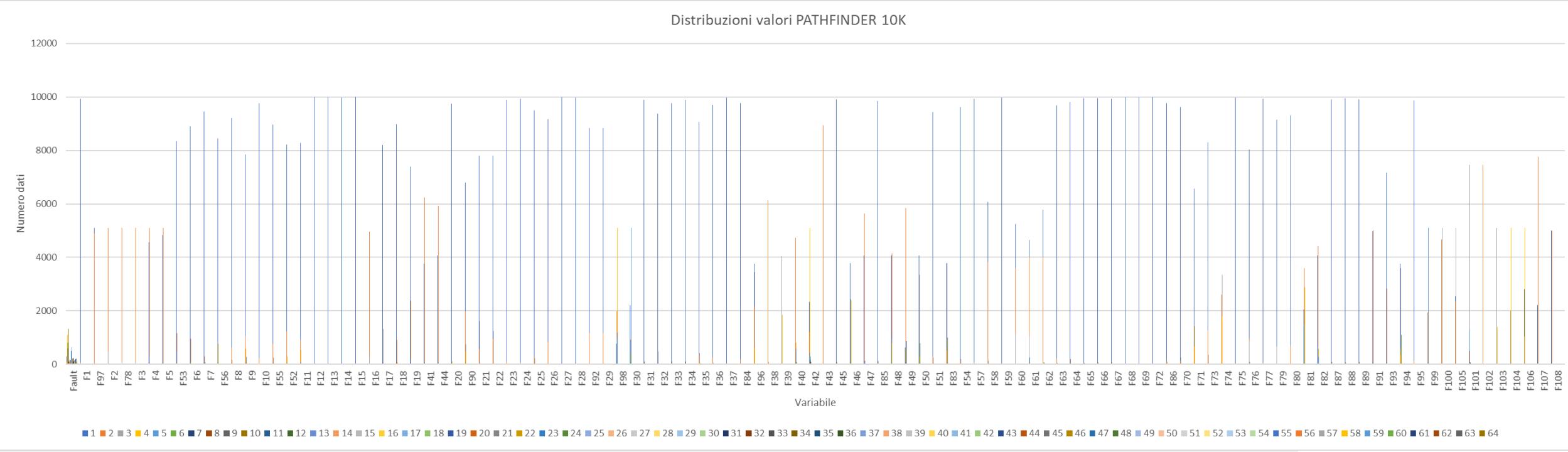
La rete conta quindi:

- 88 variabili
- 138 archi
- 912 diversi parametri
- In-degree massimo 6
- Out-degree massimo: 6
- Numero di foglie: 8
- Numero di radici: 14

# INTRODUZIONE AI TEST – PATHFINDER

*Una rete tradizionale molto grande, anche se meno popolare delle due precedenti. Essa è stata progettata per assistere i patologi chirurgici nella diagnosi delle malattie linfonodali.*

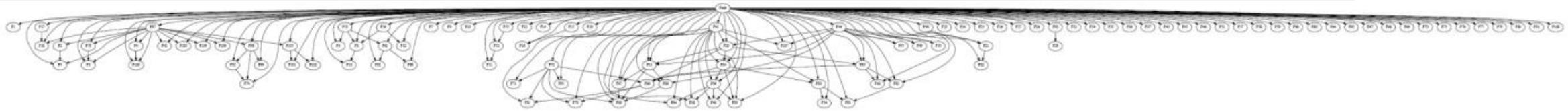
Il dataset si compone di 109 variabili, la maggior parte delle variabili sono binarie, è presente un lieve sbilanciamento dei valori. La variabile *fault* contiene 64 valori diversi. All'interno del dataset sono presenti diverse celle missing.



# INTRODUZIONE AI TEST – PATHFINDER

Per gestire il problema dei valori missing nativi, si è deciso di rimuovere dal dataset PATHFINDER 100K tutti i record che esibivano almeno un valore mancante. L'esecuzione degli algoritmi EM HARD e EM SOFT è stato effettuato su un dataset che non conteneva nessun valore missing nativo all'interno dello stesso.

# INTRODUZIONE AI TEST – PATHFINDER



La rete conta quindi:

- 109 variabili
- 195 archi
- 71890 diversi parametri
- In-degree massimo 5

# INTRODUZIONE AI TEST – Sperimentazioni effettuate

Per ogni dataset, sono state effettuate le seguenti sperimentazioni:

Dataset	Descrizione	Prop (percentuale missing values)	Repliche	Numero dati
Sports	Random patterns	0,05	10	100, 200, 400, 800, 1200, 1600, <b>5000</b>
		0,1	10	100, 200, 400, 800, 1200, 1600
	Patterns sui nodi più rilevanti	0,05	10	100, 200, 400, 800, 1200, 1600, 2000
		0,1	10	100, 200, 400, 800, 1200, 1600
Property	Random patterns	0,01	8	200, 400, 800, 1100
		0,05	8	400, 800, 1100
	MCAR	0,01	8	200, 400, 800, 1100
	Nodi più connessi	0,01	8	200, 400, 800, 1100
	Foglie	0,01	8	200, 400, 800, 1100
ForMed	Random patterns	0,006	8	300, 600, 1000, 1400
		0,01	8	300, 600, 1000, 1400
	Radici	0,003	8	300, 600, 1000, 1400
	Con maggiore outdegree	0,003	8	300, 600, 1000, 1400
	Foglie	0,006	8	300, 600, 1000, 1400
	MCAR	0,006	8	300, 600, 1000, 1400
	Nodi più connessi	0,006	8	300, 600, 1000, 1400
Pathfinder	Random patterns	0,005	8	300,600,1000, 1400
		0,01	8	1000
	Nodi più connessi	0,005	8	300,600,1000, 1400
	Con maggiore indegree	0,005	8	300,600,1000
	Con maggiore outdegree	0,005	8	300,600,1000
	Foglie	0,005	8	300,600,1000
	MCAR	0,005	8	300,600,1000

# INTRODUZIONE AI TEST – Sperimentazioni effettuate

Definizioni:

- **Patterns sui nodi più rilevanti:** i patterns vengono definiti in maniera casuale, generando missing values per i soli nodi foglia e i nodi più connessi (con almeno 3 archi entranti o uscenti). Tale definizione è applicata solo su reti di piccole dimensioni come SPORTS.
- **Nodi foglia:** I missing values sono generati principalmente su quei nodi che non hanno alcun arco uscente.
- **MCAR:** I missing values sono generati in modo tale che essi risultano essere generati in modo completamente random.
- **Più connessi:** Viene calcolata una top-5 per le reti di medie dimensioni (Property) e una top-10 per le reti di grandi dimensioni (ForMed e PathFinder) sulla base del numero di archi entranti uscenti. Se due o più nodi condividono lo stesso grado, tutti i nodi vengono selezionati.
- iodvjkhdjiofhvjisvhnioprehvjckoshniopwehroghu

# INTRODUZIONE AI TEST – Sperimentazioni effettuate

Si danno ora le definizioni per dataset valido

Random patterns, MCAR e Patterns sui nodi più rilevanti

Fissiamo:

$$\begin{aligned}\text{Numero missing values ideali (NMVI)} &= \# \text{ celle} * \text{prop} \\ \text{Tolleranza di errore (E)} &= \text{NMVI} * \text{error\_rate}\end{aligned}$$

*Un dataset risulta essere **valido** se il numero totale di missing values risulta appartenere al seguente intervallo [NMVI - E, NMVI + E]*

**Esempio:**

Supponiamo che numero totale di celle in un subset di sports sia uguale a 9.000, fissando  $\text{prop} = 0,01$  e  $\text{error\_rate} = 0,1$ , un dataset è valido se il numero totale di celle missing rientra nell'intervallo: [81,99]

# INTRODUZIONE AI TEST – Sperimentazioni effettuate

Si danno ora le definizioni per dataset valido

Nodi più connessi, Foglie, Radici, con maggiore in-degree/out-degree

Fissiamo:

**Numero missing values ideali (NMVI) = # celle \* prop**

**Tolleranza di errore (E) = NMVI \* error\_rate**

*Un dataset risulta essere **valido** se il numero totale di missing values risulta appartenere al seguente intervallo [NMVI - E, NMVI + E], e, per i nodi selezionati, la percentuale di missing values è maggiore o uguale di 0,9*

**Esempio:**

Supponiamo che numero totale di celle in un subset di Property sia uguale a 10.000, fissando *prop* = 0,01 e *error\_rate* = 0,1, un dataset è valido se il numero totale di celle missing rientra nell'intervallo: [90,110].

Supponiamo di aver generato un dataset con 100 missing values e che siamo intenzionati a generare i missing values sulle foglie della rete, Siano A, B e C i nodi foglia. **Un dataset è valido se il numero di missing values in corrispondenza dei nodi A, B e C è maggiore o uguale di 90.**

# INTRODUZIONE AI TEST – Sperimentazioni effettuate

Per ogni sperimentazione, è stata computata la **Kullback-Leibler divergence**, già affrontata nelle presentazioni precedenti. Tuttavia è importante evidenziare come la **ground truth** viene estratta

## INTRODUZIONE AI TEST – Sperimentazioni effettuate

La **ground truth** è stata ricavata dalla versione 100K di ciascun dataset, attraverso la funzione *fit* messa a disposizione da **bnlearn**.

I subset sono stati ricavati dalla versione 10K di ciascun dataset per motivi di efficienza computazione.

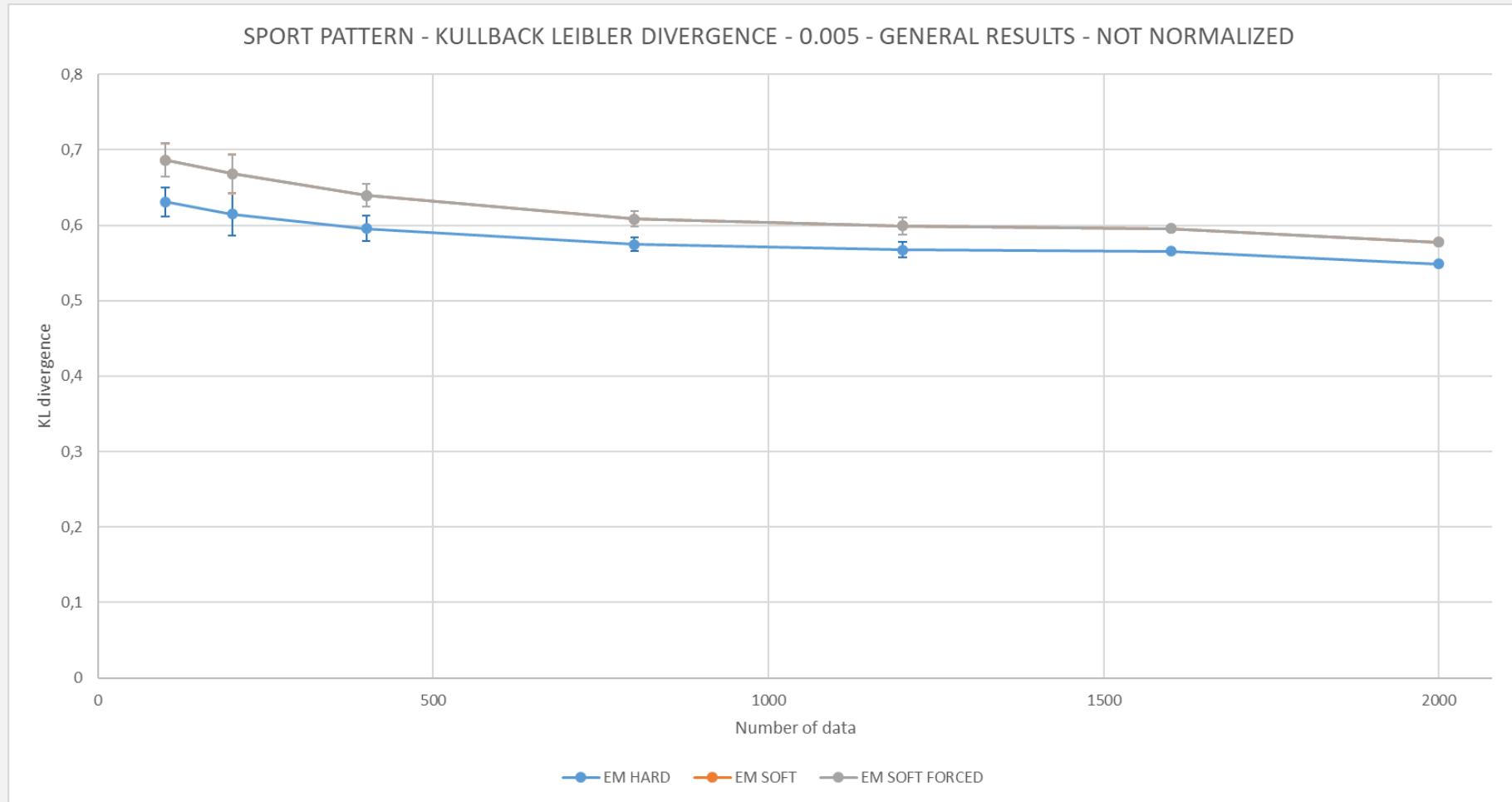
Alla fine della computazione, sono stati estratti i risultati, in termini di KL divergence, sia in termini generali sull'intera rete, sia su ciascun nodo coinvolto nell'analisi.

# SPERIMENTAZIONI E RISULTATI

*Sports  
Property  
ForMed  
Pathfinder*

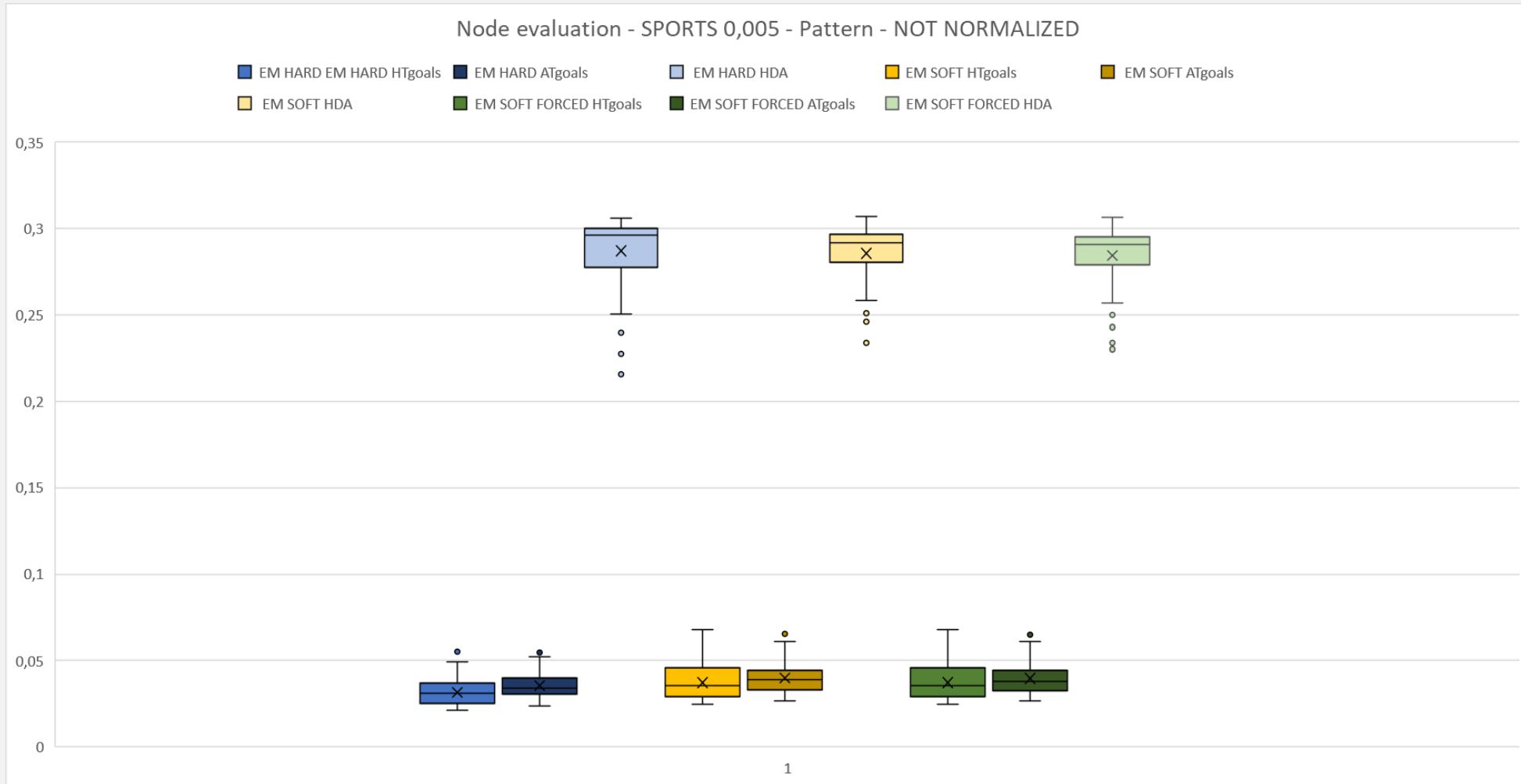
# Sperimentazioni - SPORTS

*Sports – Patterns sui nodi più rilevanti – 0.05*



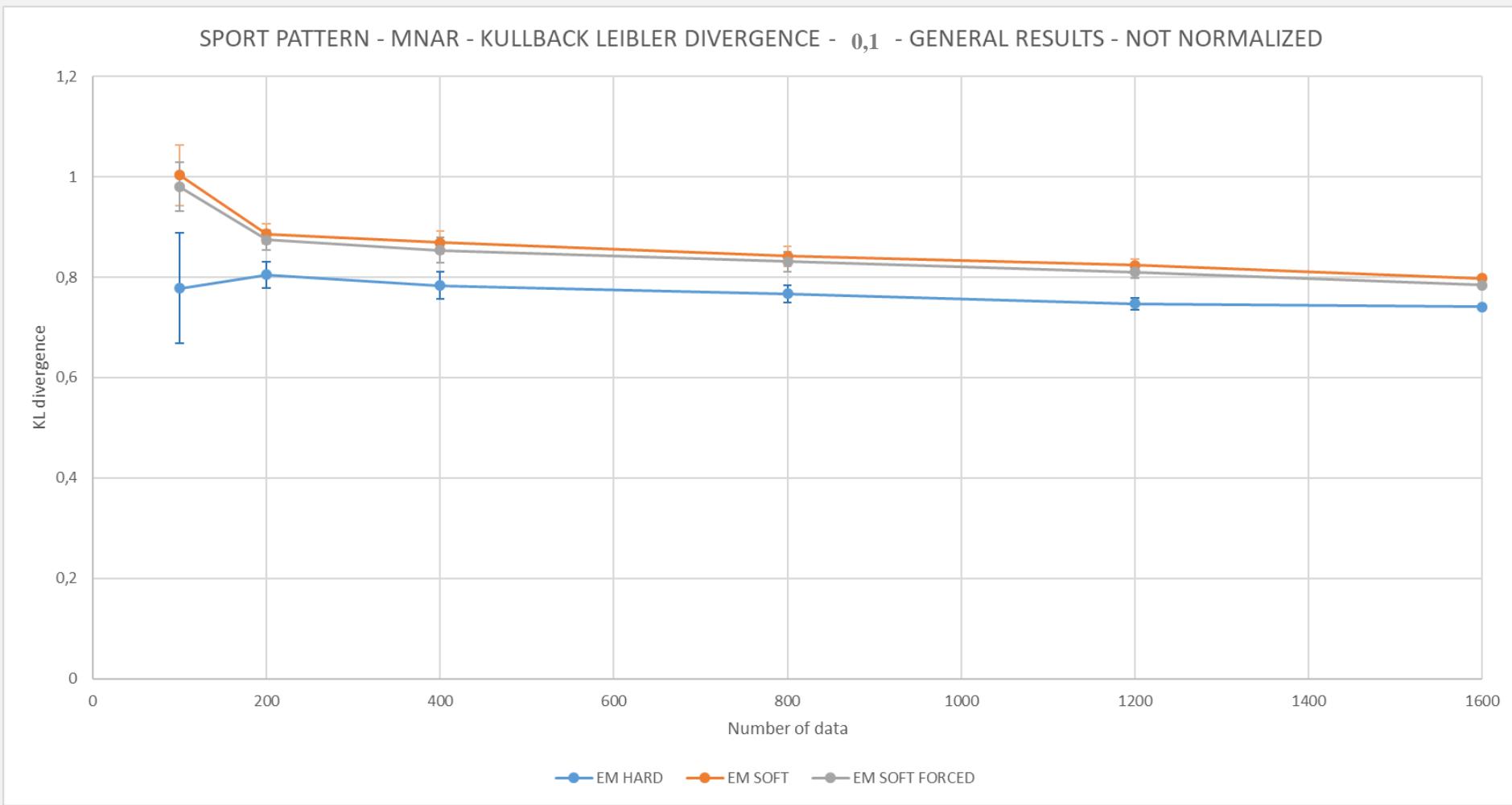
# Sperimentazioni - SPORTS

*Sports – Patterns sui nodi più rilevanti – 0.05*



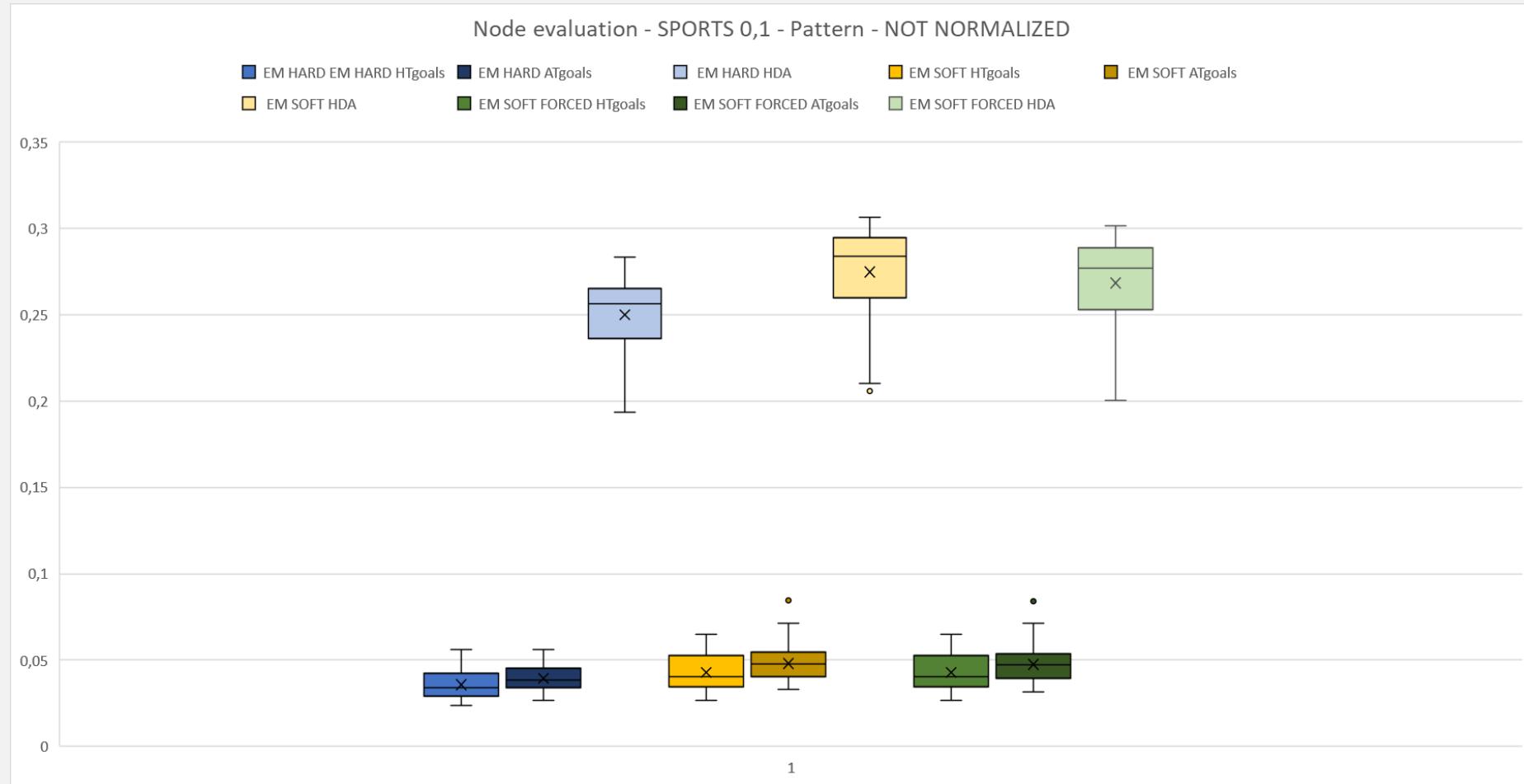
# Sperimentazioni - SPORTS

*Sports – Patterns sui nodi più rilevanti – 0.1*



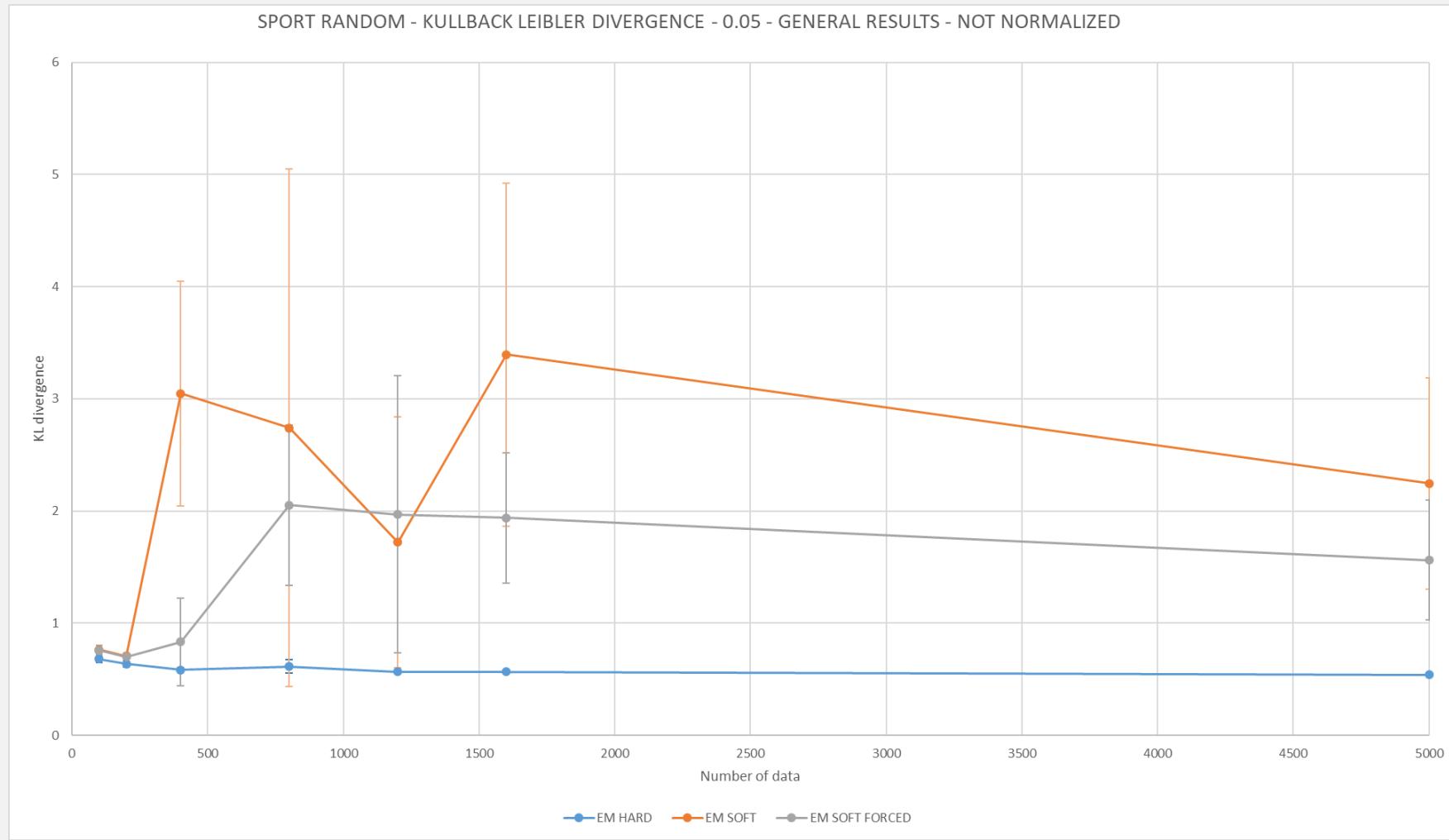
# SPERIMENTAZIONI - SPORTS

*Sports – Patterns sui nodi più rilevanti – 0.1*



# Sperimentazioni - SPORTS

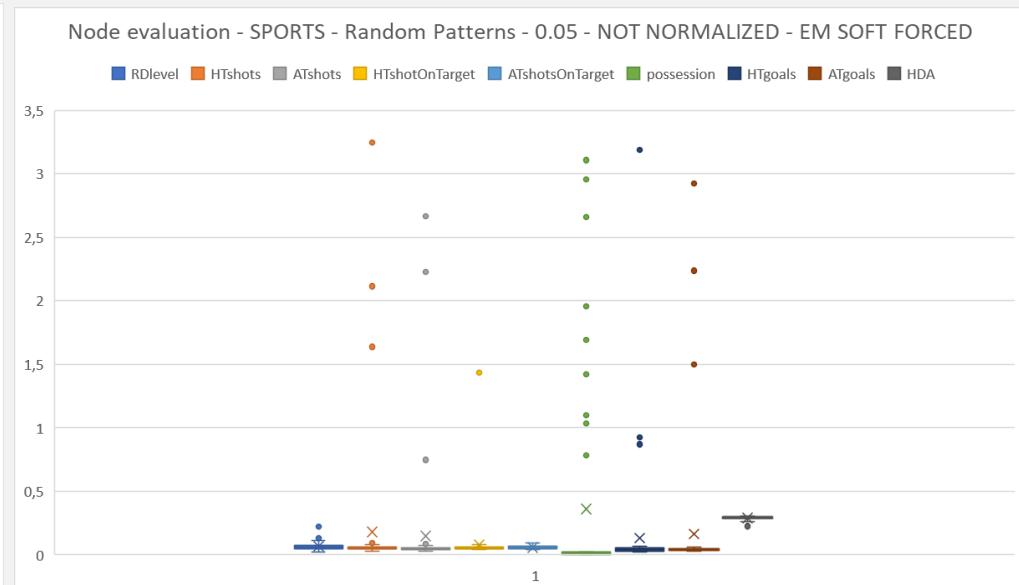
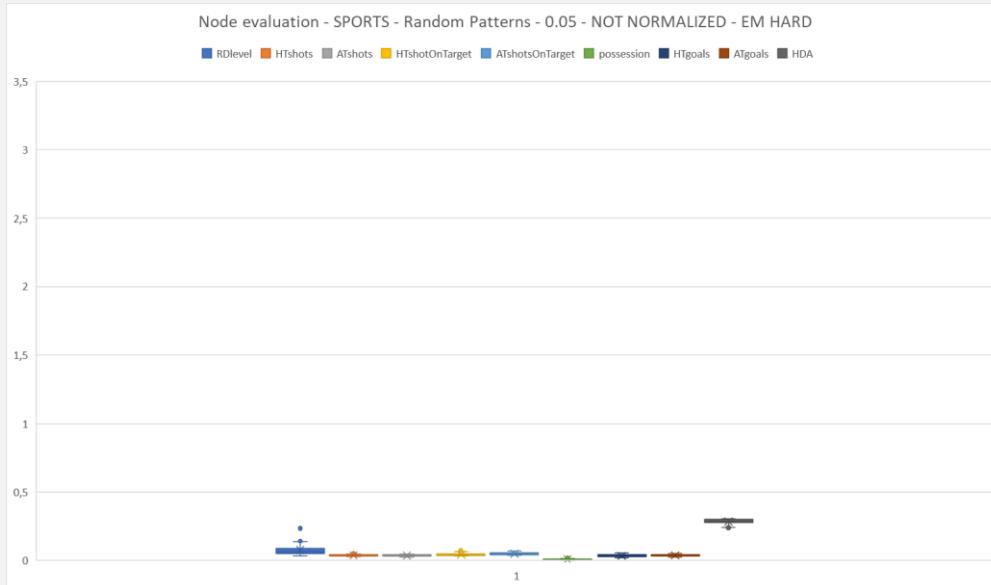
*Sports – Random patterns – 0.05*



# SPERIMENTAZIONI - SPORTS

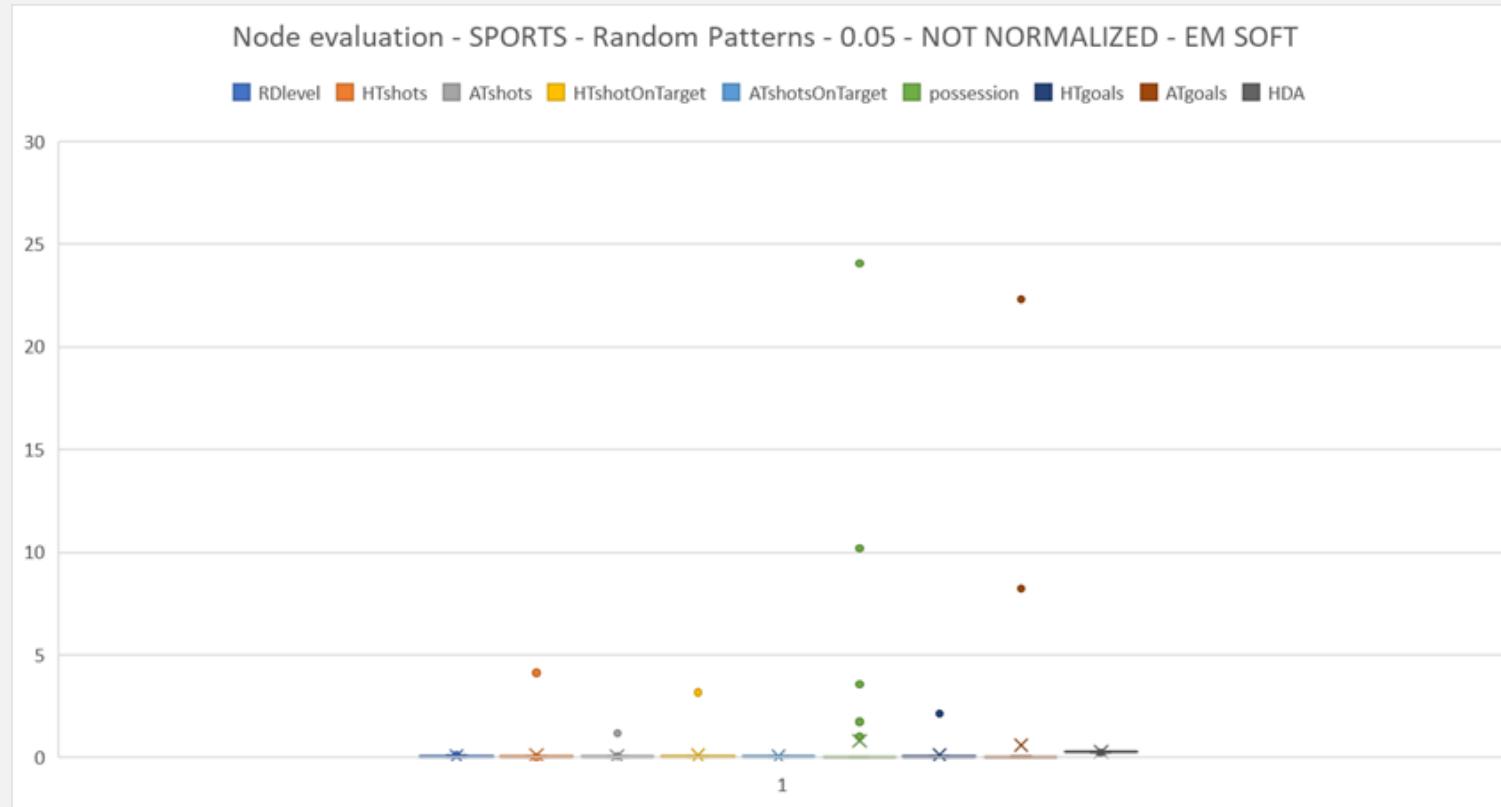


*Sports – Random patterns – 0.05*



# SPERIMENTAZIONI - SPORTS

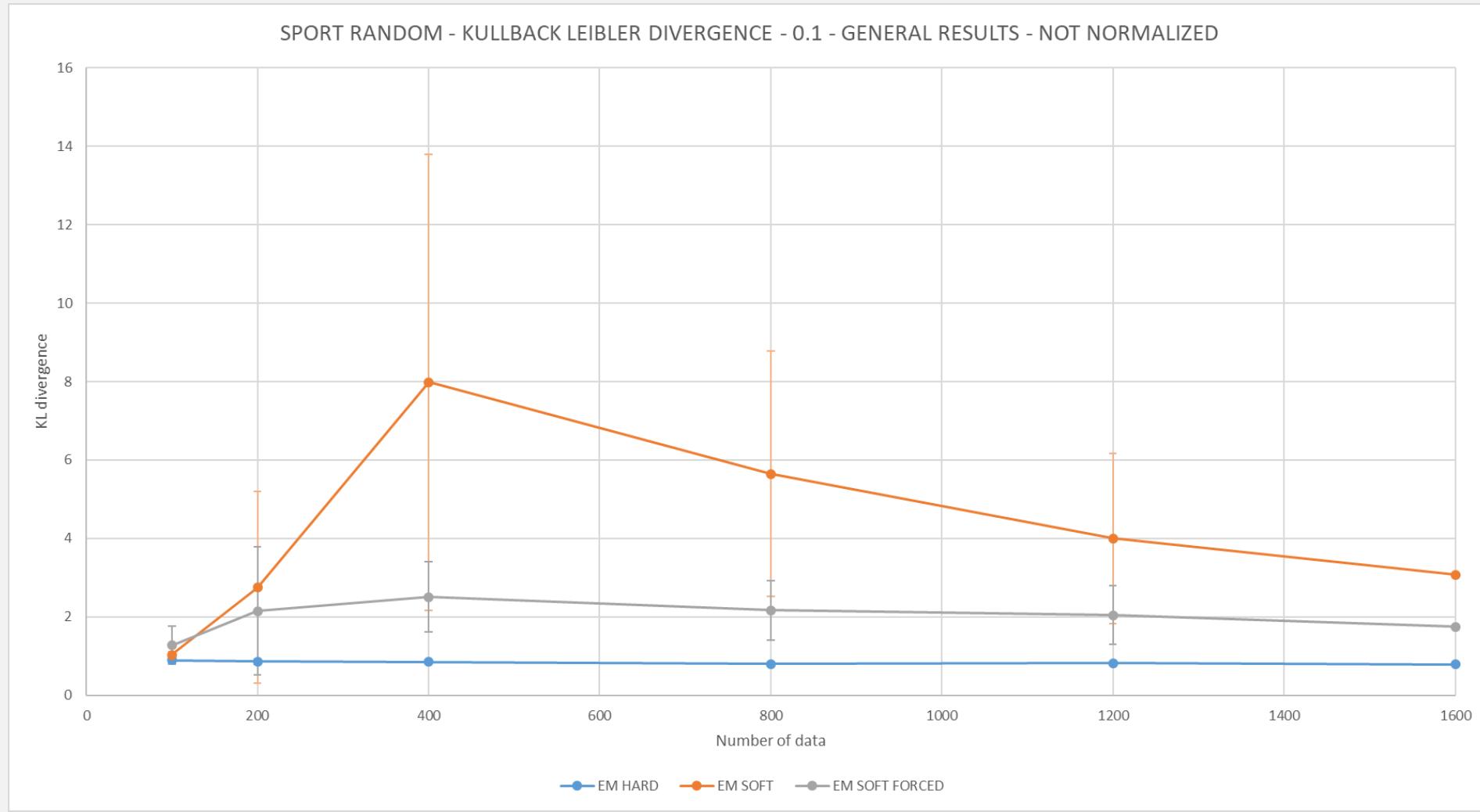
*Sports – Random patterns – 0.05*



N.B Attenzione alla diversa scala rispetto alla slide precedente

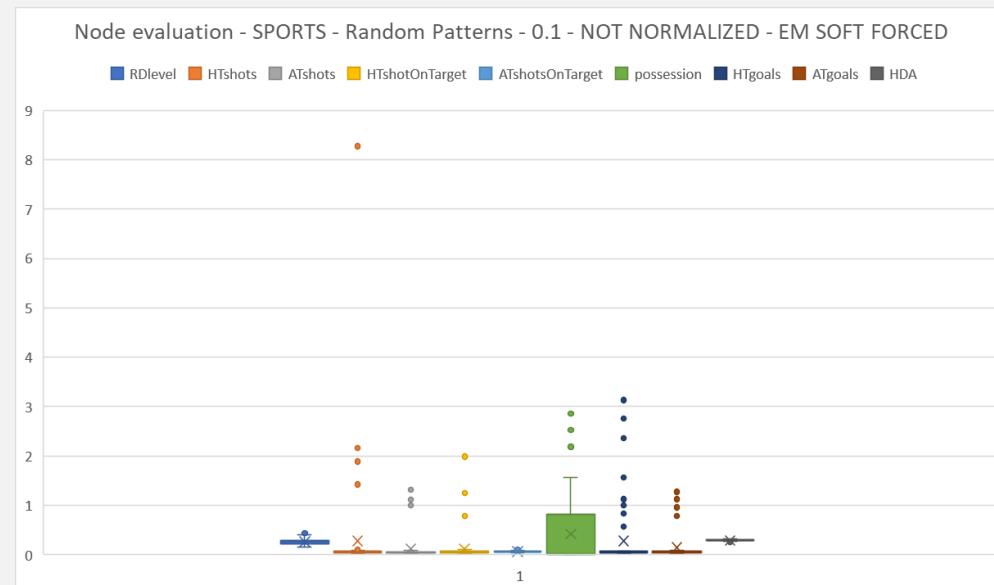
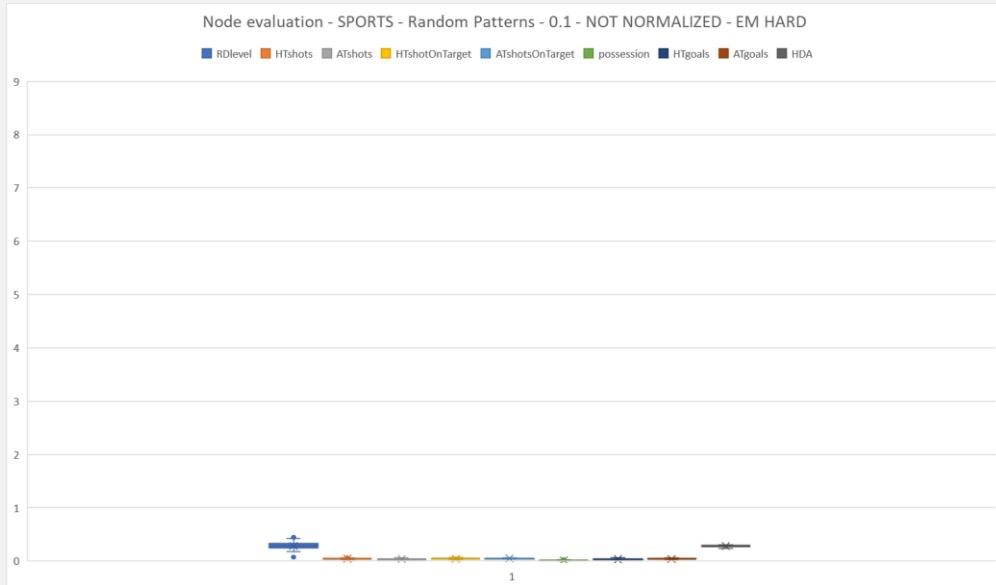
# Sperimentazioni - SPORTS

*Sports – Random patterns – 0.1*



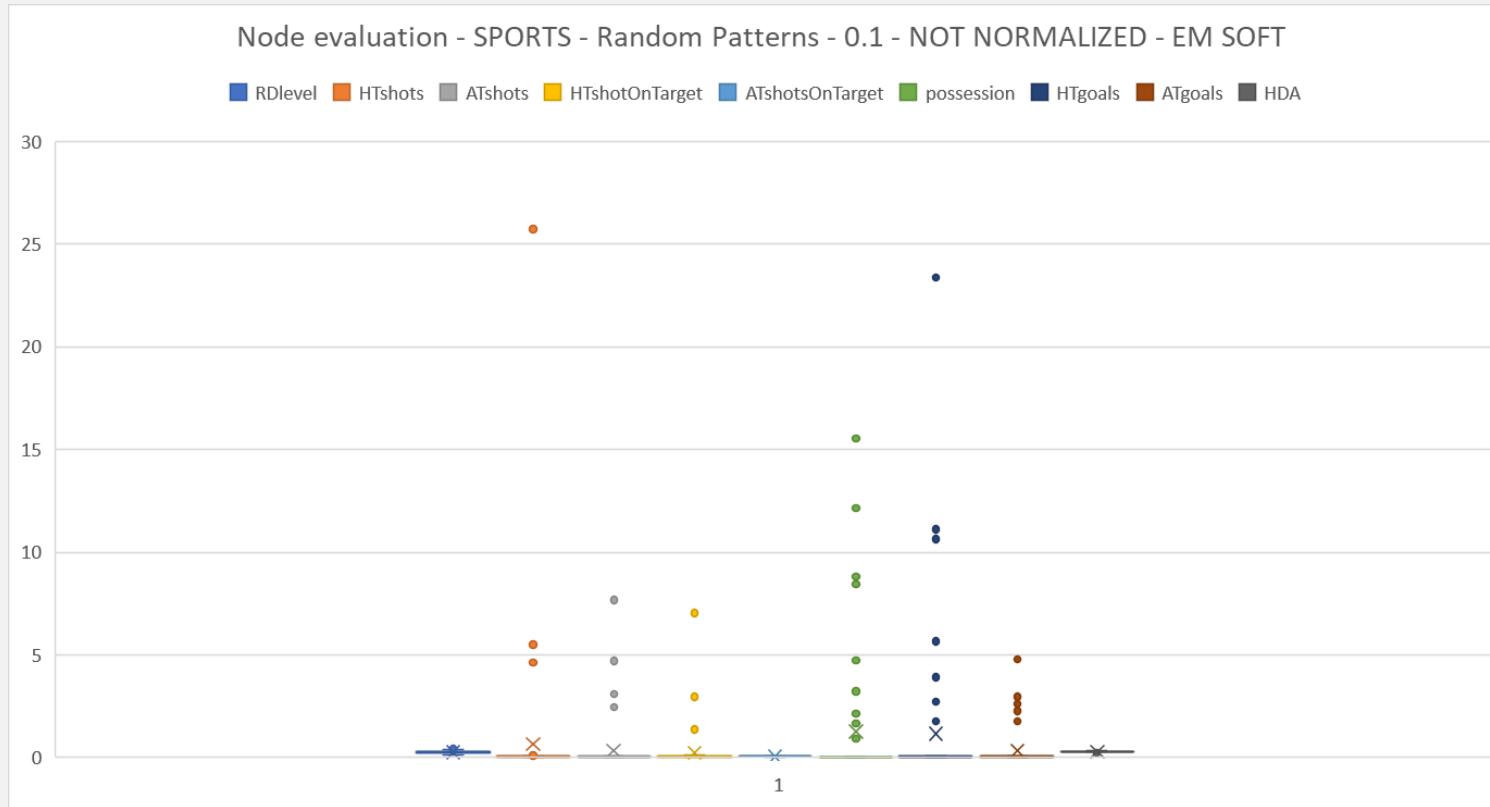
# Sperimentazioni - SPORTS

*Sports* – Random patterns – 0.1



# SPERIMENTAZIONI - SPORTS

*Sports – Random patterns – 0.1*

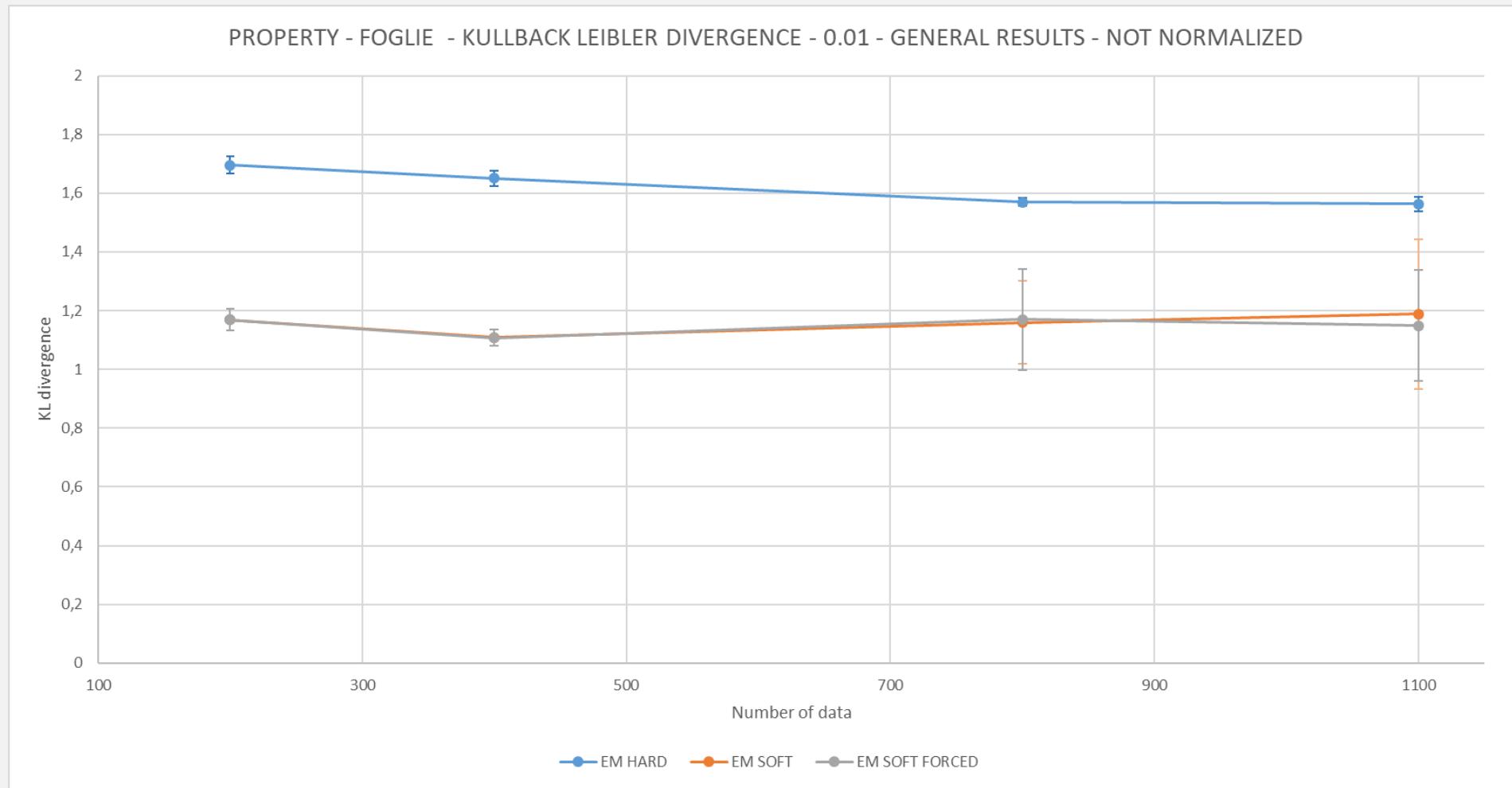


N.B Attenzione alla diversa scala rispetto alla slide precedente

# SPERIMENTAZIONI - PROPERTY

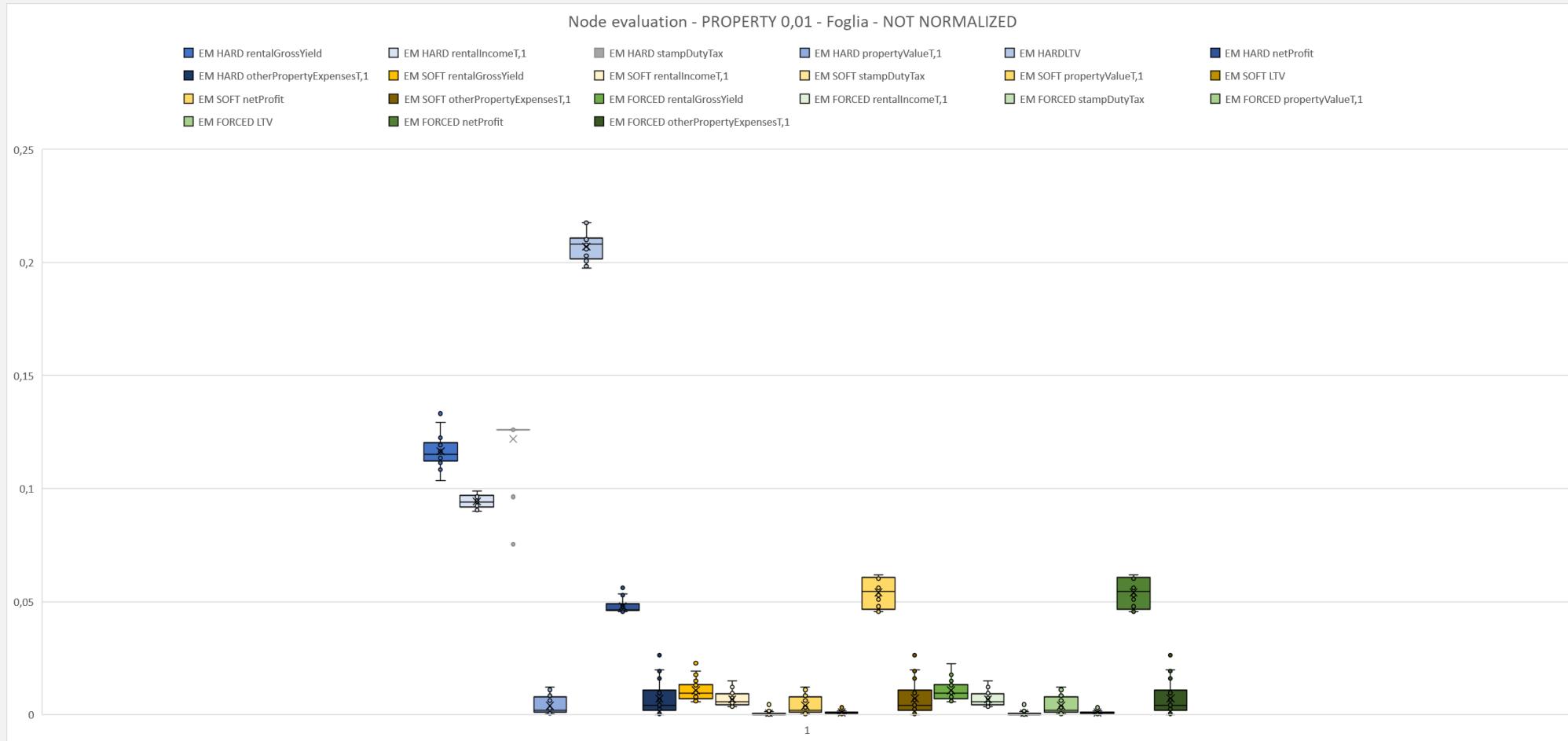


*Property – Foglie – 0.01*



# SPERIMENTAZIONI - PROPERTY

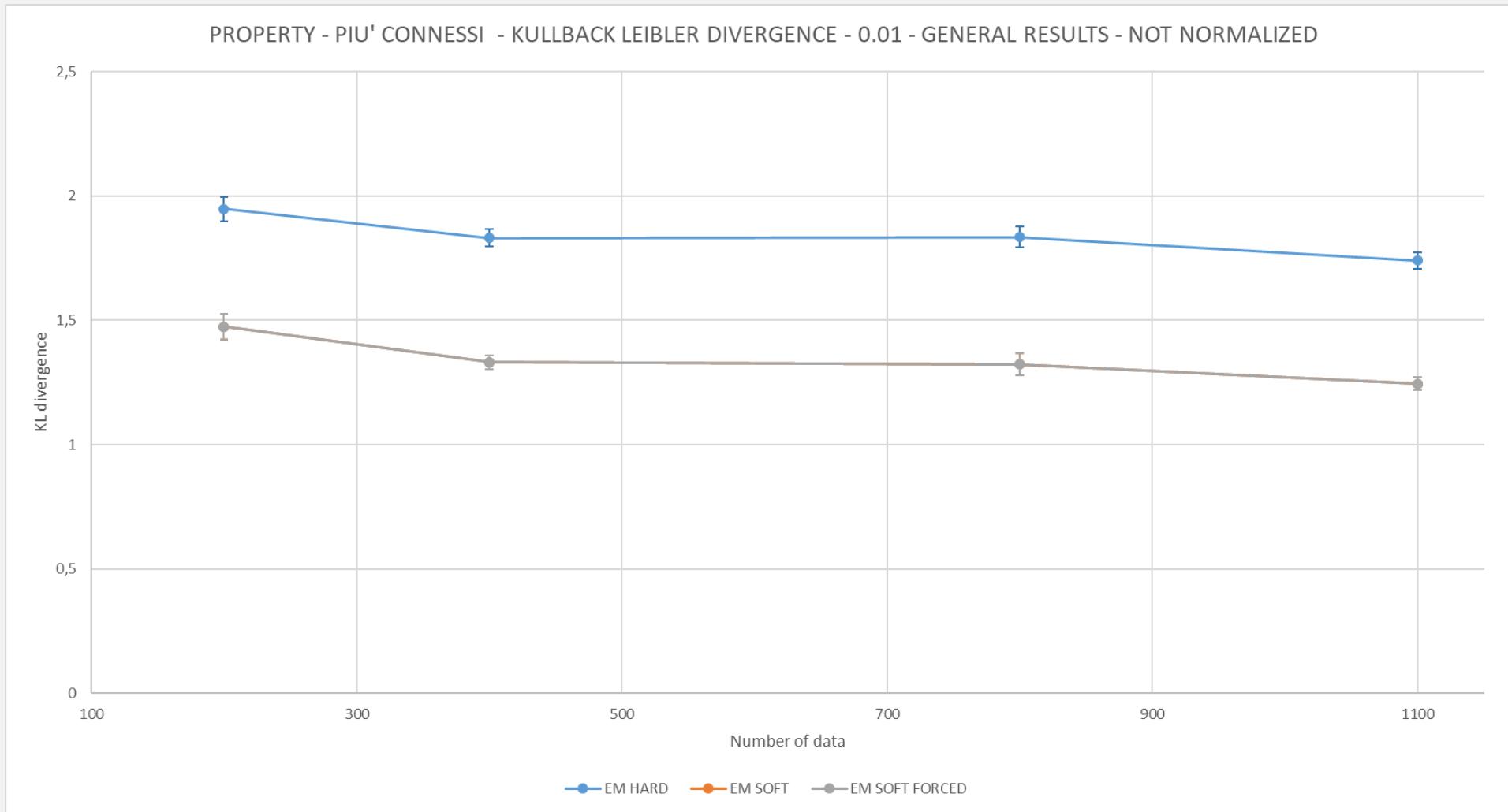
Sports – Foglie – 0.01



# SPERIMENTAZIONI - PROPERTY



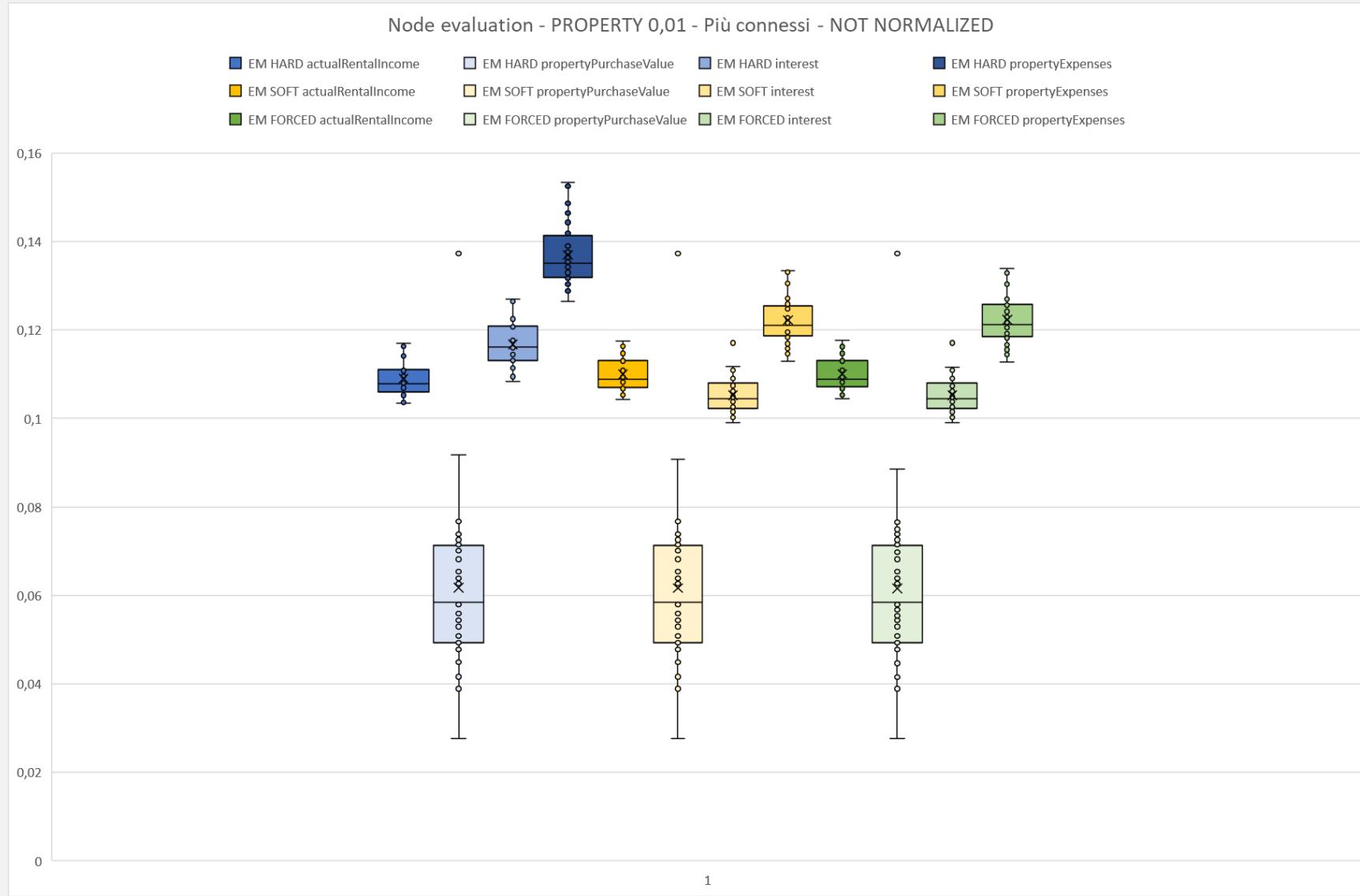
*Property – PIU' CONNESSI – 0.01*



N.B da notare che i valori della KL divergence risultano essere più alti rispetto al caso precedente (foglie)

# SPERIMENTAZIONI - PROPERTY

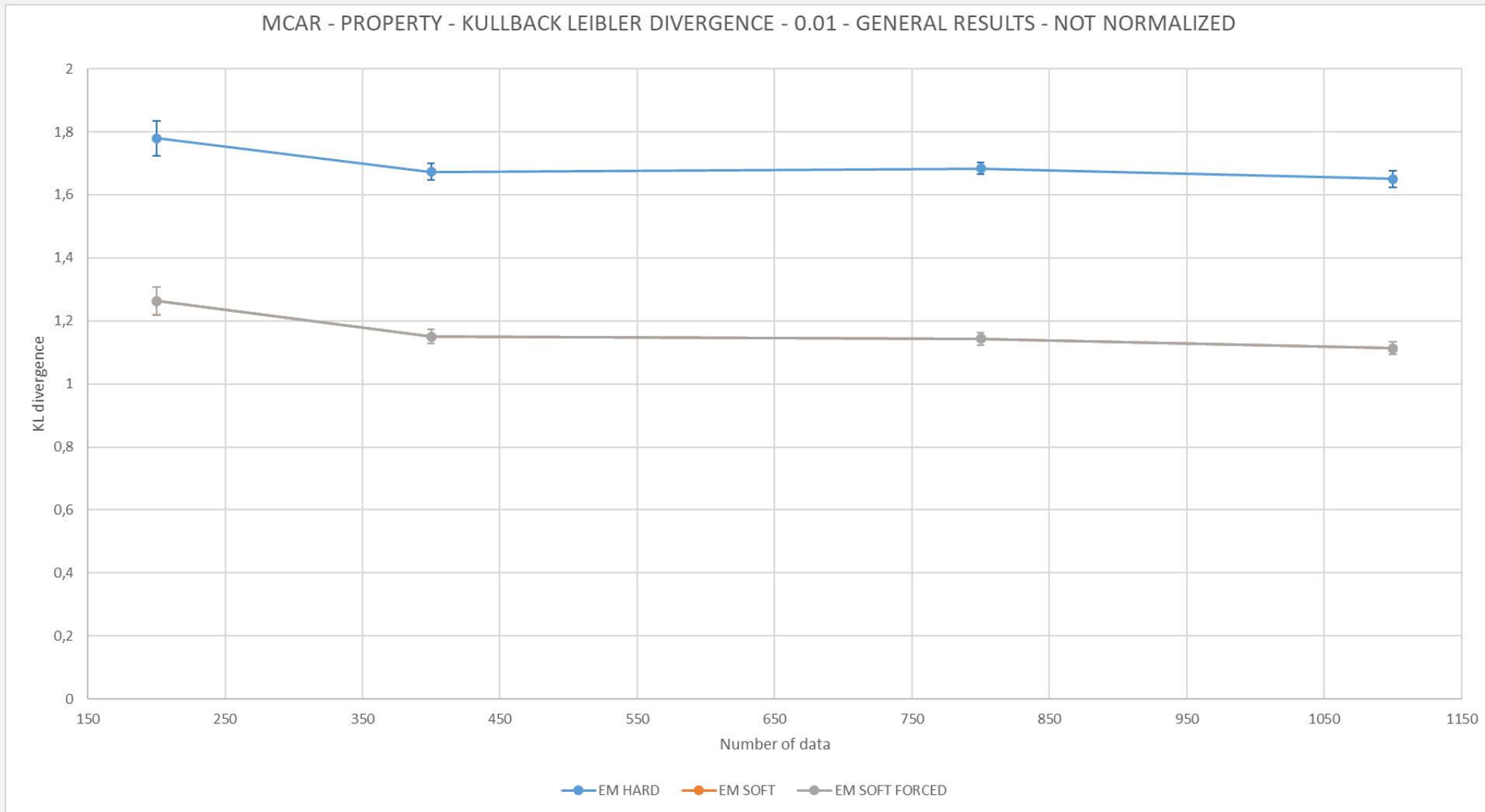
## Property – PIU' CONNESSI – 0.01



# SPERIMENTAZIONI - PROPERTY



*Property – MCAR – 0.01*



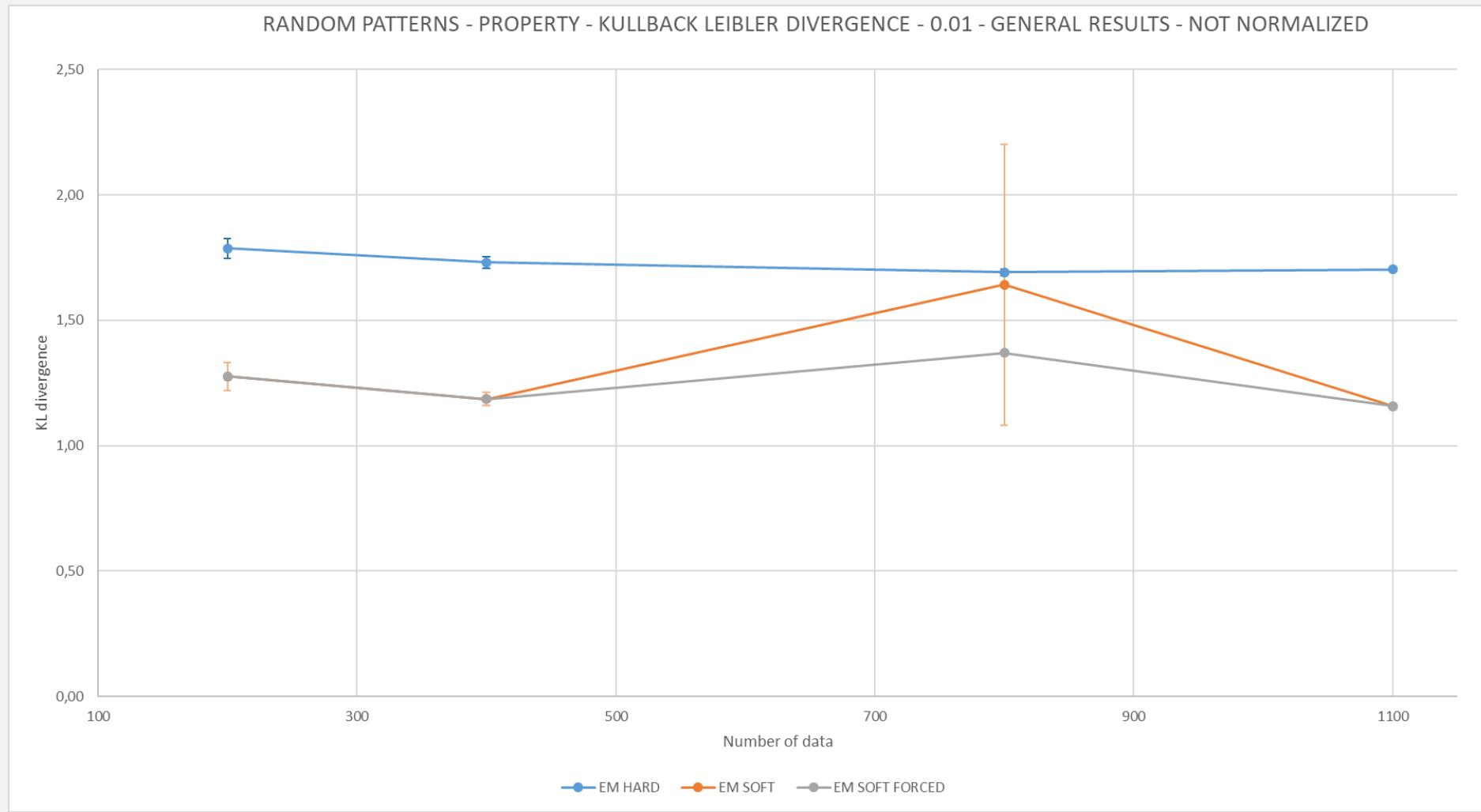
# SPERIMENTAZIONI - PROPERTY

## Property – MCAR – 0.01



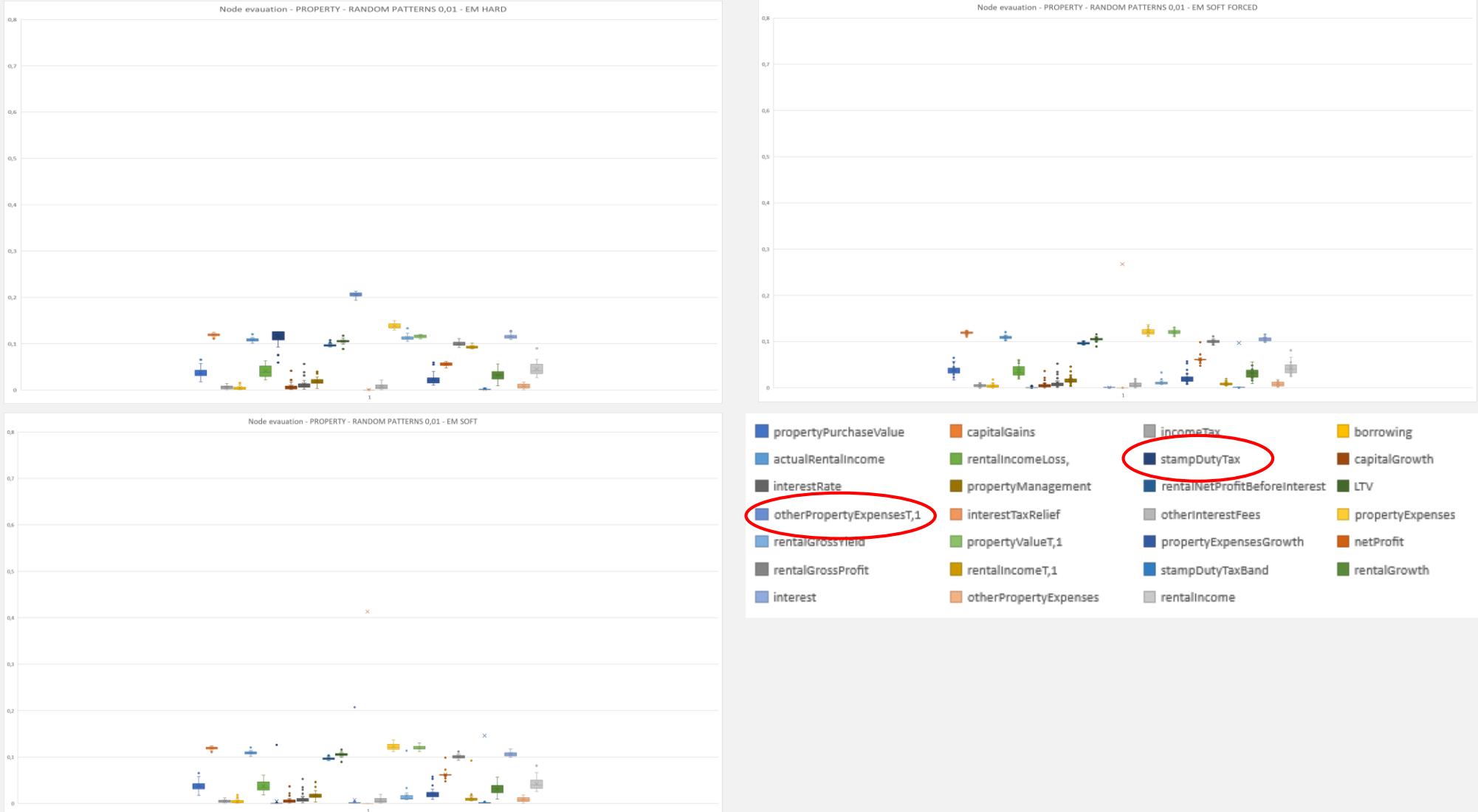
# SPERIMENTAZIONI - PROPERTY

*Property – Random patterns – 0.01*



# SPERIMENTAZIONI - PROPERTY

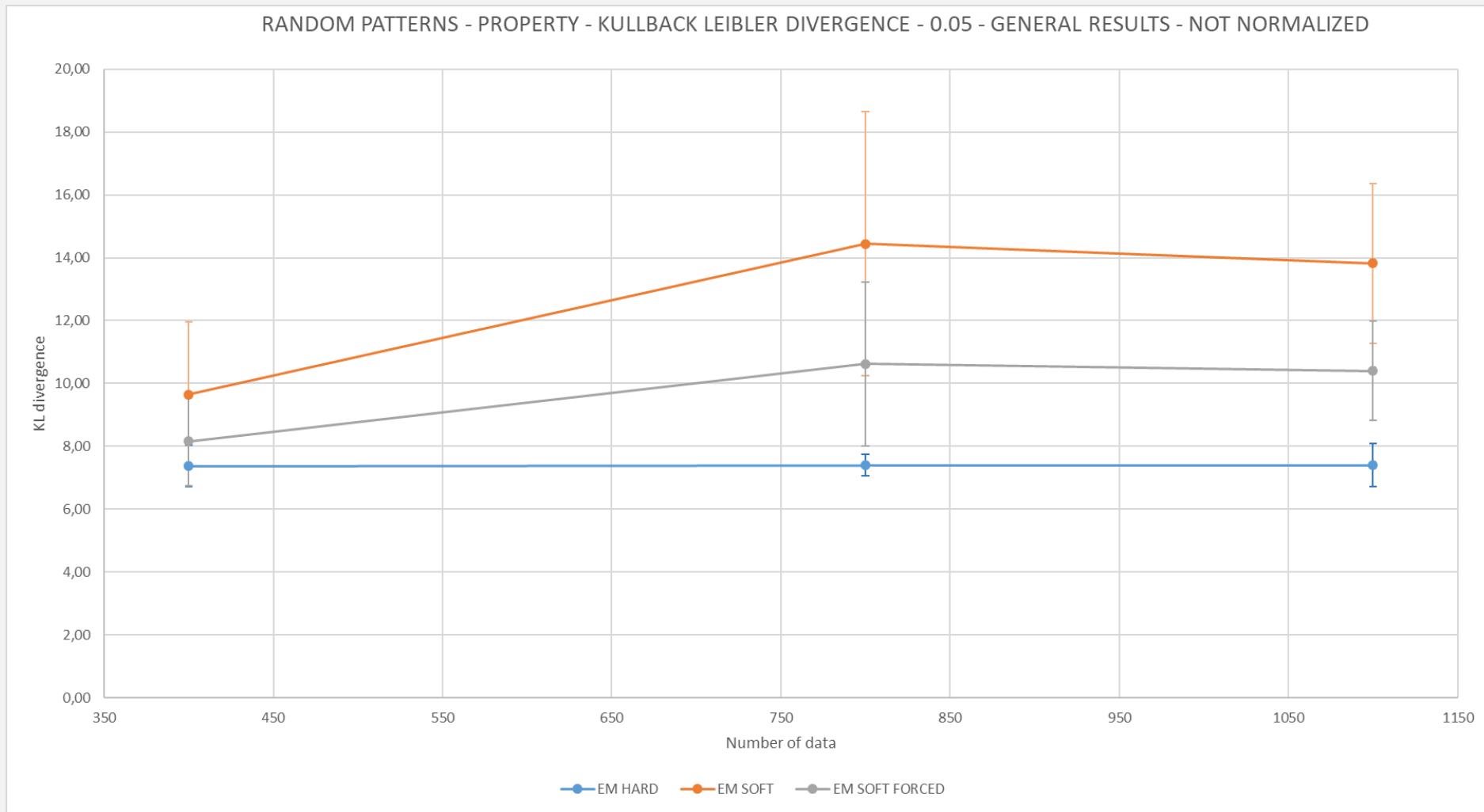
## *Property* – Random patterns – 0.01



# SPERIMENTAZIONI - PROPERTY



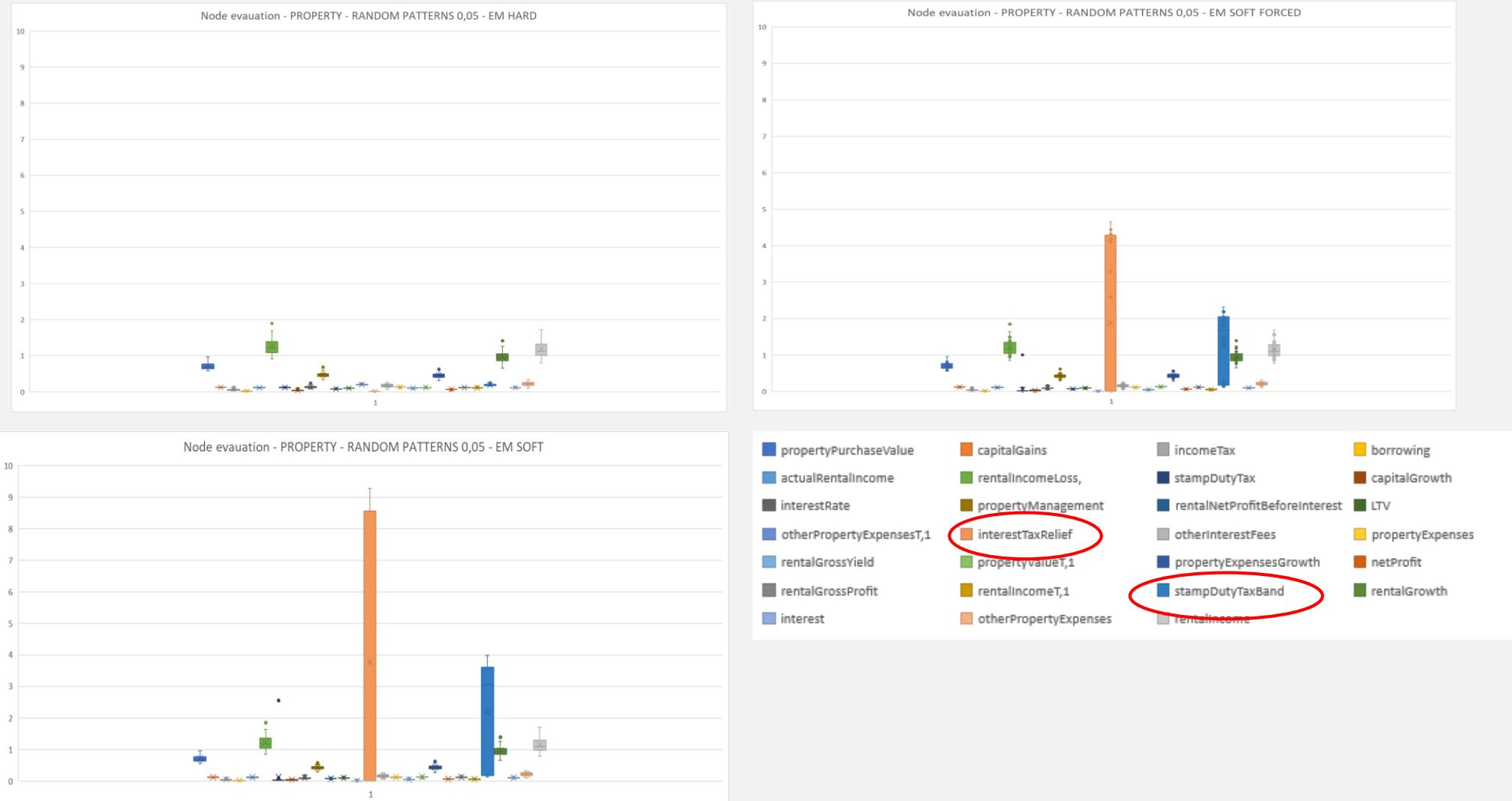
*Property – Random patterns – 0.05*



N.B da notare che  
all'aumentare  
dell'iperparametro  
prop, EM HARD, in  
questo caso, diventa  
significativamente  
migliore

# SPERIMENTAZIONI - PROPERTY

## Property – Random patterns – 0.05



# SPERIMENTAZIONI - PROPERTY

## Property – Random patterns – 0.05

Prima di esaminare i restanti dataset, si vogliono indagare le cause che hanno condotto ai risultati appena presentati.

**Osservazione 1 :** EM HARD ha concluso la computazione alla **terza** iterazione, EM SOFT FORCED alla **quarta** iterazione, mentre, EM SOFT alla **sesta** iterazione.

**Osservazione 2:** Osservando le distribuzioni dei valori delle variabili *interestTaxRelief* e *stampDutyTaxBand* ci si accorge che le variabili risultano avere probabilità uniforme per ciascun valore. La prima variabile può assumere 7 valori diversi, mentre la seconda 6.

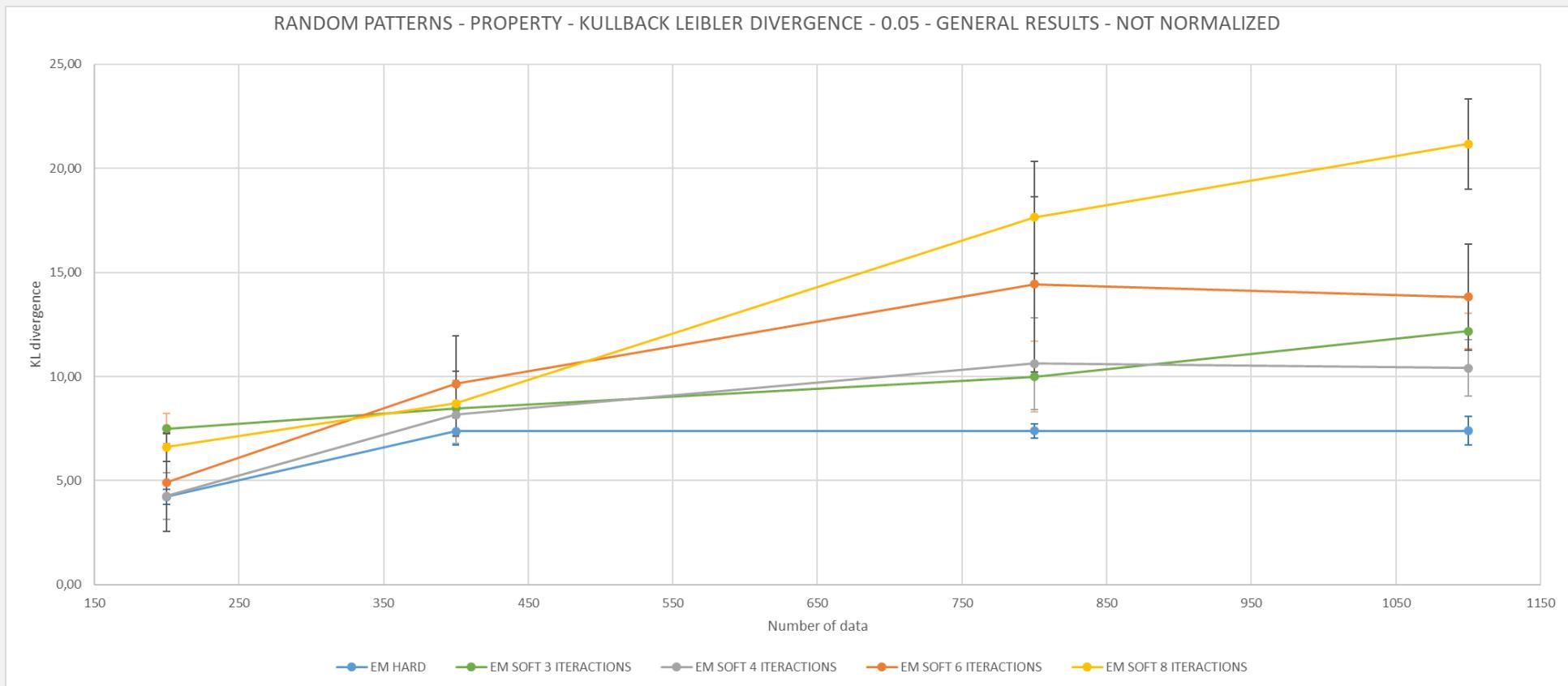
**Ipotesi:** **overfitting** impostando il numero di iterazioni molto alto, EM SOFT impara troppo bene la distribuzione di probabilità uniforme delle variabili e in alcune situazioni, combina “disastri” quando deve rimpiazzare i valori mancanti, come visualizzato nei risultati mostrati precedentemente. Si tratta quindi di un fenomeno di overfitting. Tale fenomeno viene aggravato in EM SOFT che computa ben 2 iterazioni in più rispetto alla variante FORCED.

**Setup sperimentale:** Si riesegue l'esperimento variando il numero di iterazioni, nel primo forzando il **numero di iterazioni di EM SOFT a 3**, nel secondo caso forzando il **numero di iterazioni a 8**.

**Interpretazione:** L'ipotesi di overfitting risulta essere corretta se i risultati ottenuti forzando il numero di iterazioni pari a 3, risultano essere migliori rispetto al risultato ottenuto da EM SOFT FORCED, oppure peggiori nel caso in cui l'algoritmo non sia riuscito a fittare correttamente le altre variabili. In aggiunta, si attende che i risultati ottenuti forzando il numero di iterazioni pari a 8 siano peggiori.

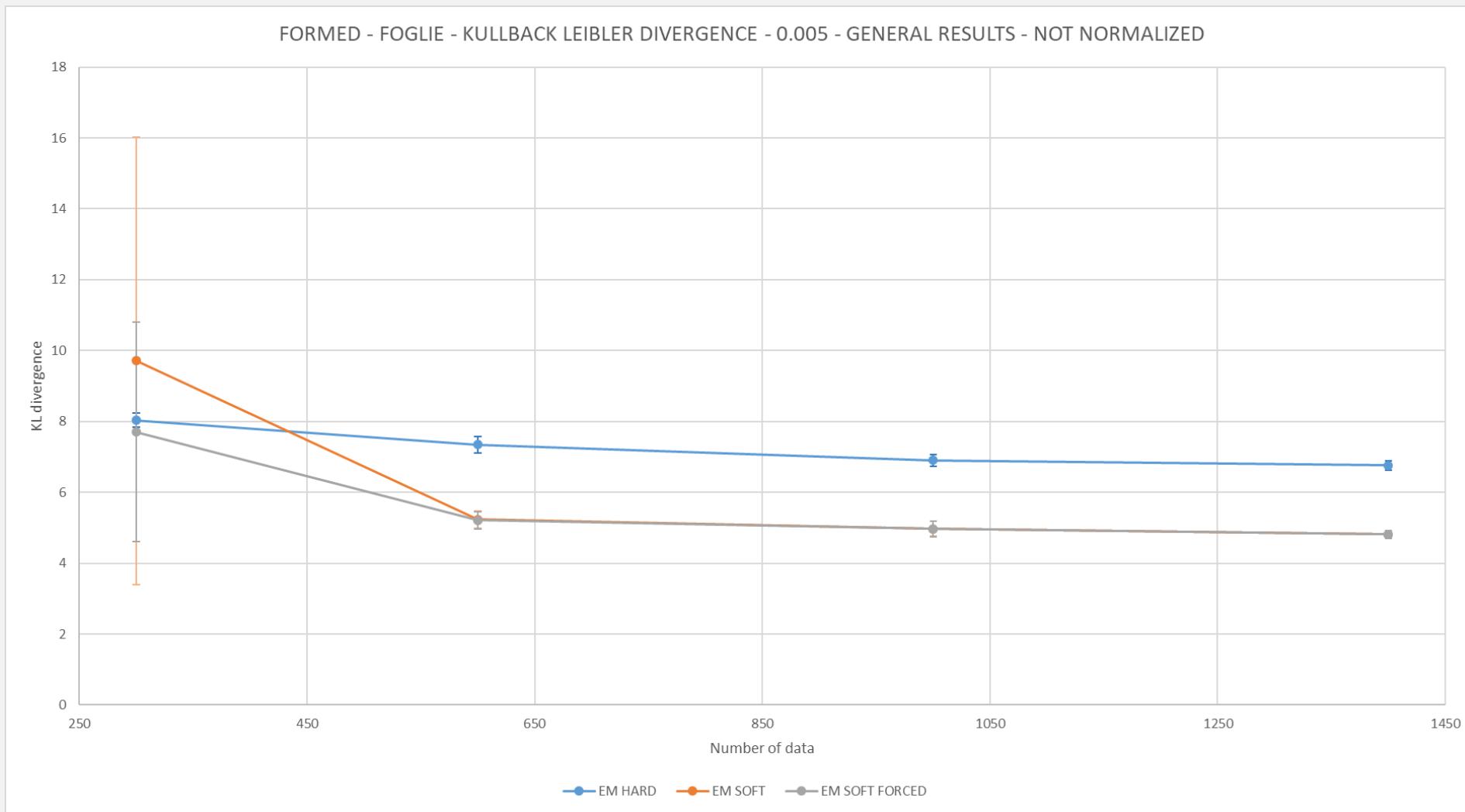
# SPERIMENTAZIONI - PROPERTY

*Property – Random patterns – 0.05*



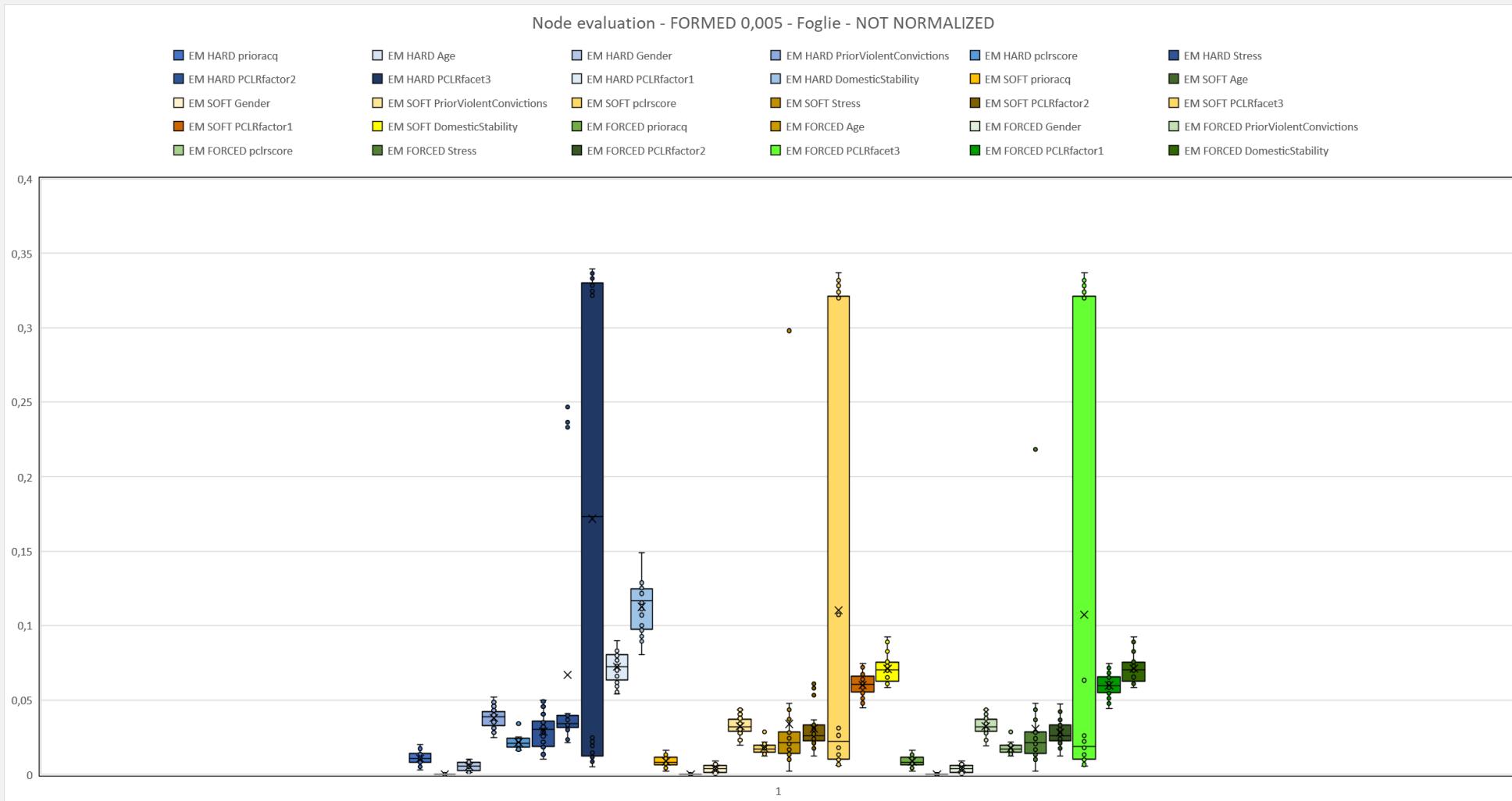
# SPERIMENTAZIONI - FORMED

## FORMED – FOGLIE – 0.005



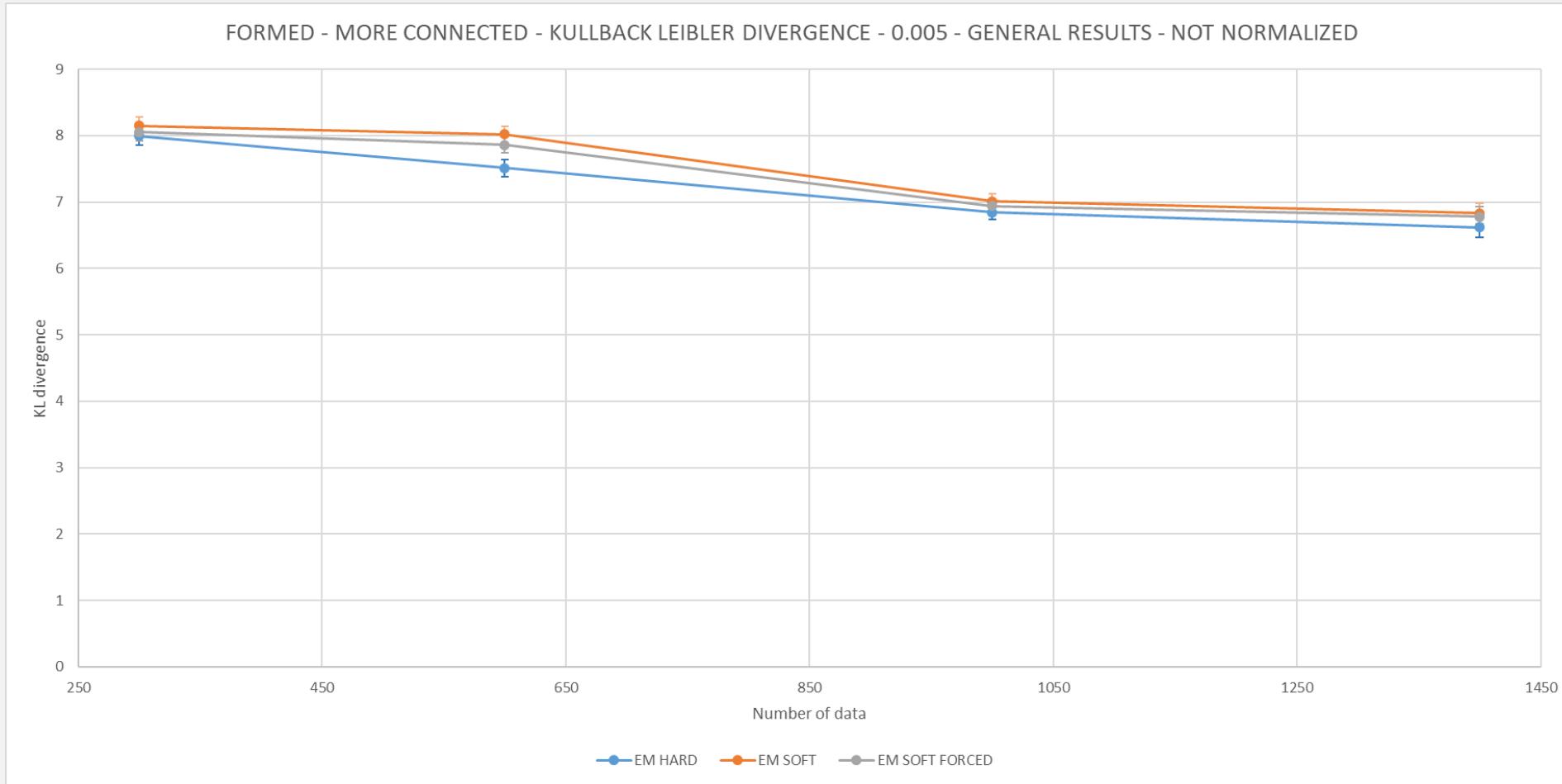
# SPERIMENTAZIONI - FORMED

## FORMED – FOGLIE – 0.005



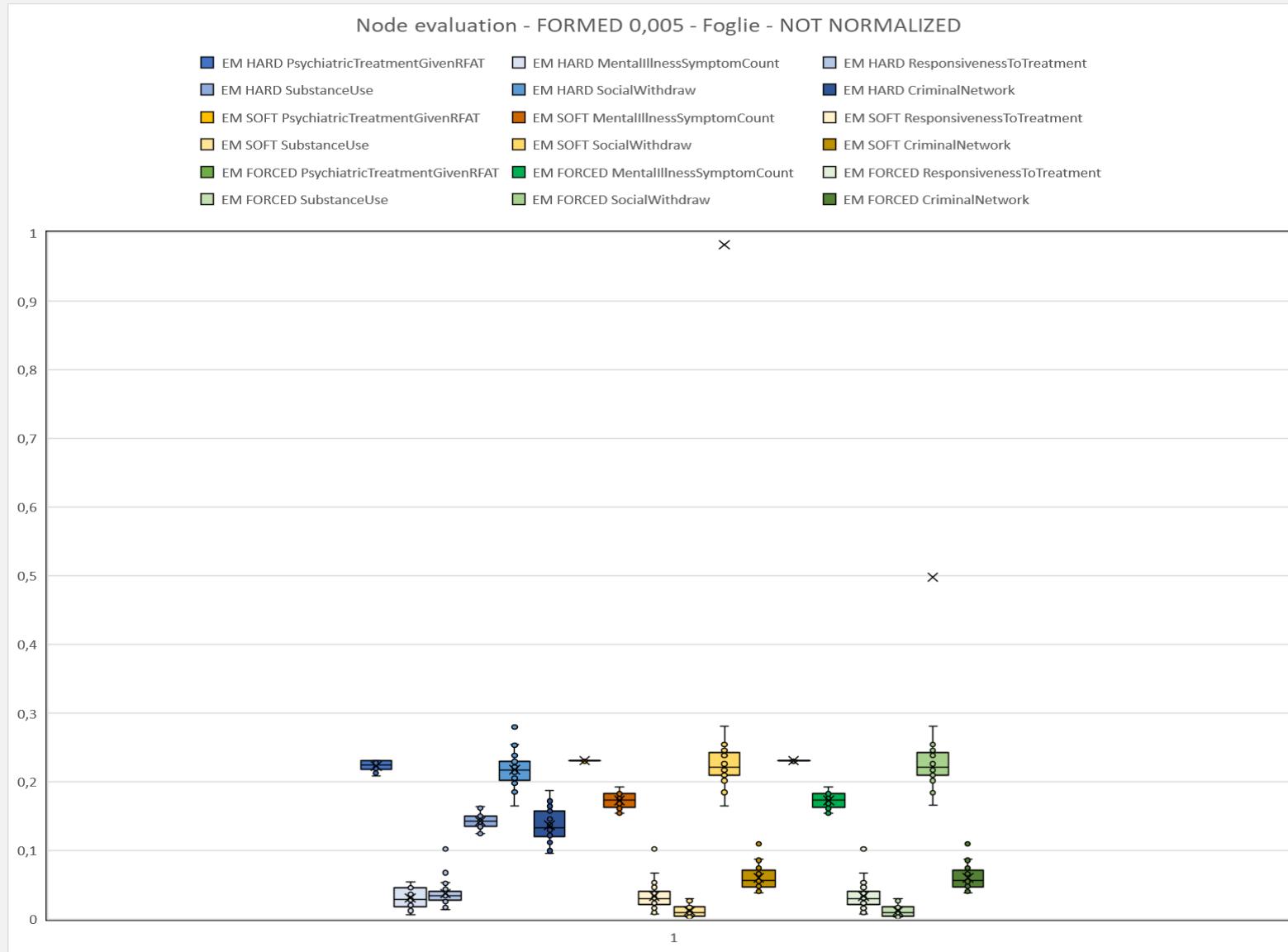
# SPERIMENTAZIONI - FORMED

FORMED – MORE CONNECTED – 0.005



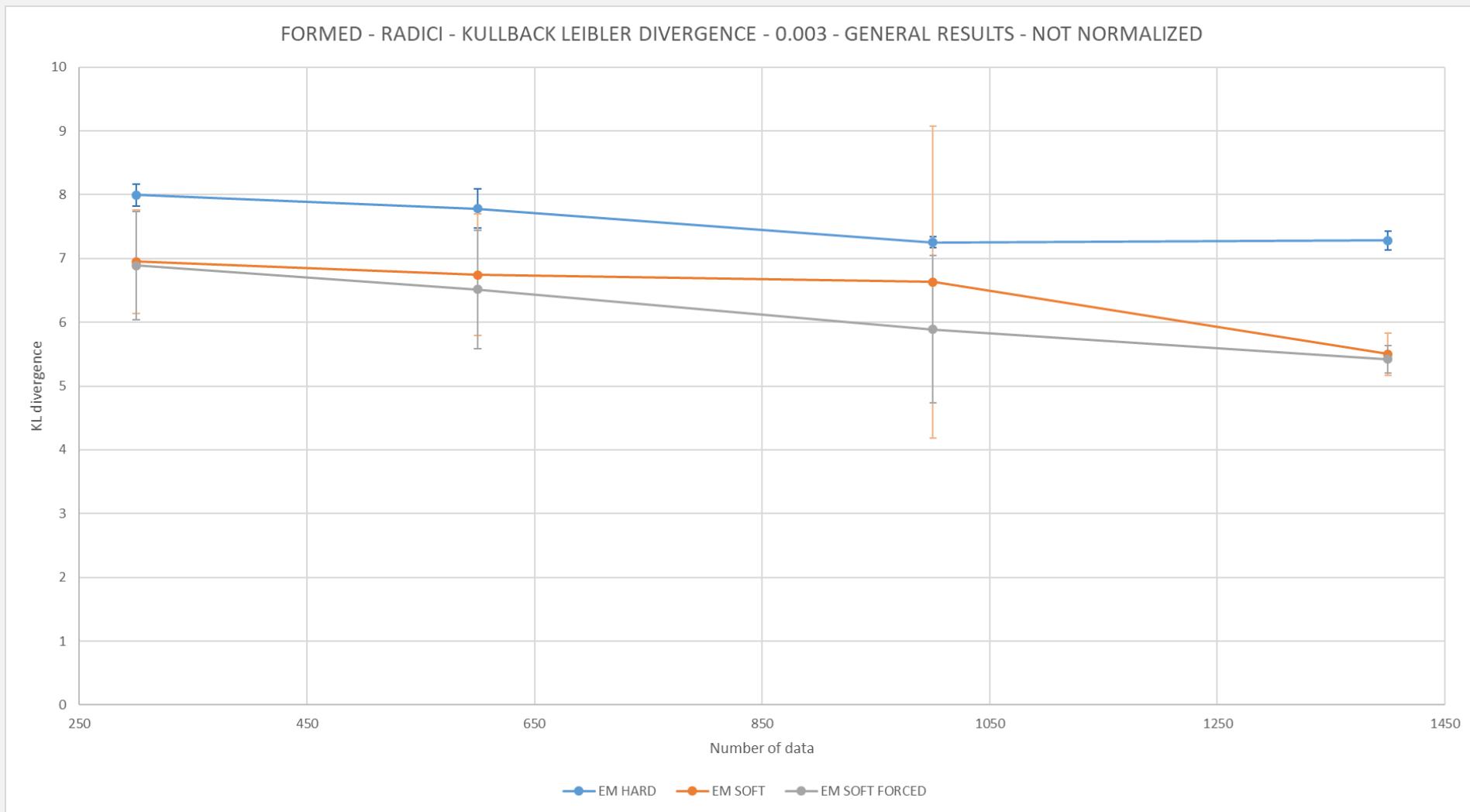
# SPERIMENTAZIONI - FORMED

## FORMED – MORE CONNECTED – 0.005



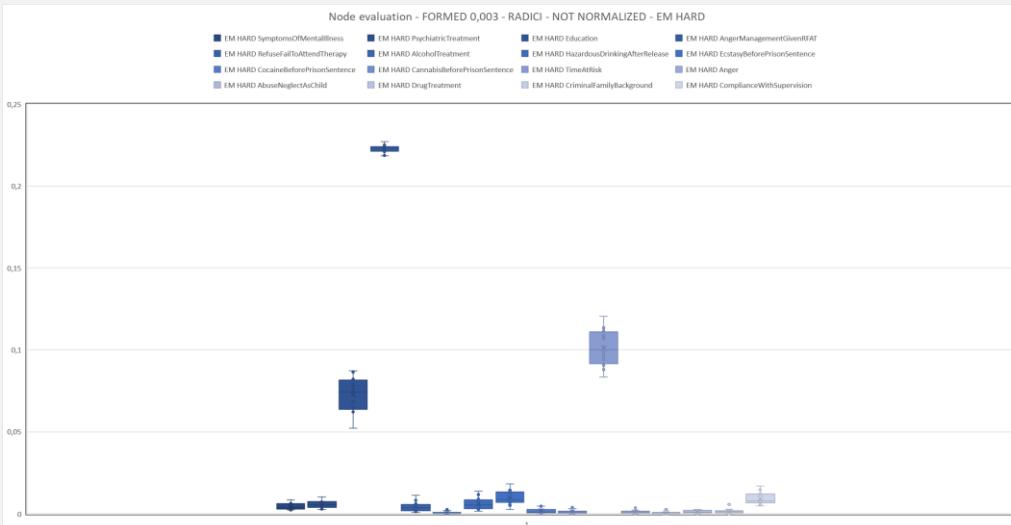
# SPERIMENTAZIONI - FORMED

FORMED – RADICI – 0.003



# SPERIMENTAZIONI - FORMED

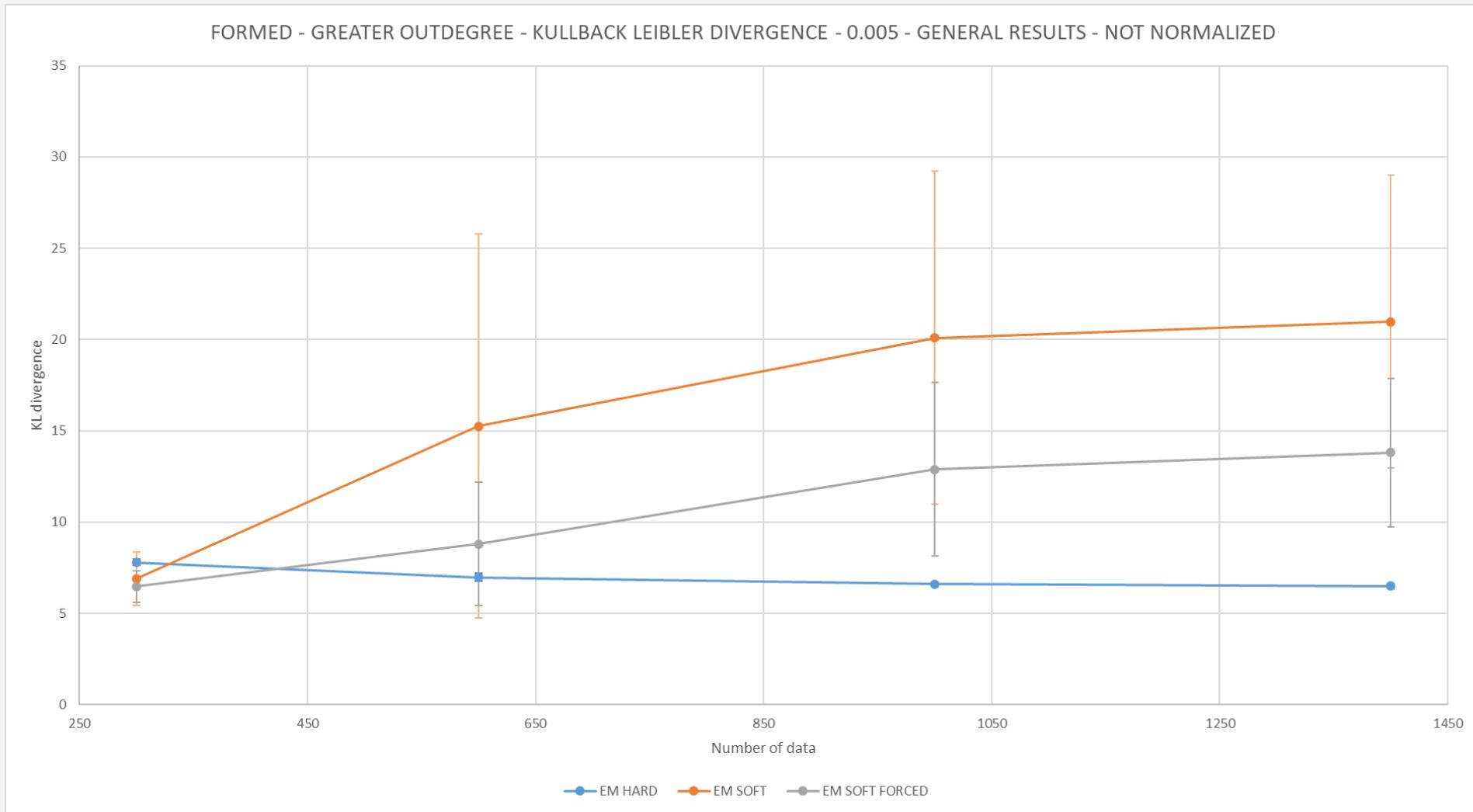
## FORMED – RADICI – 0.003



# SPERIMENTAZIONI - FORMED

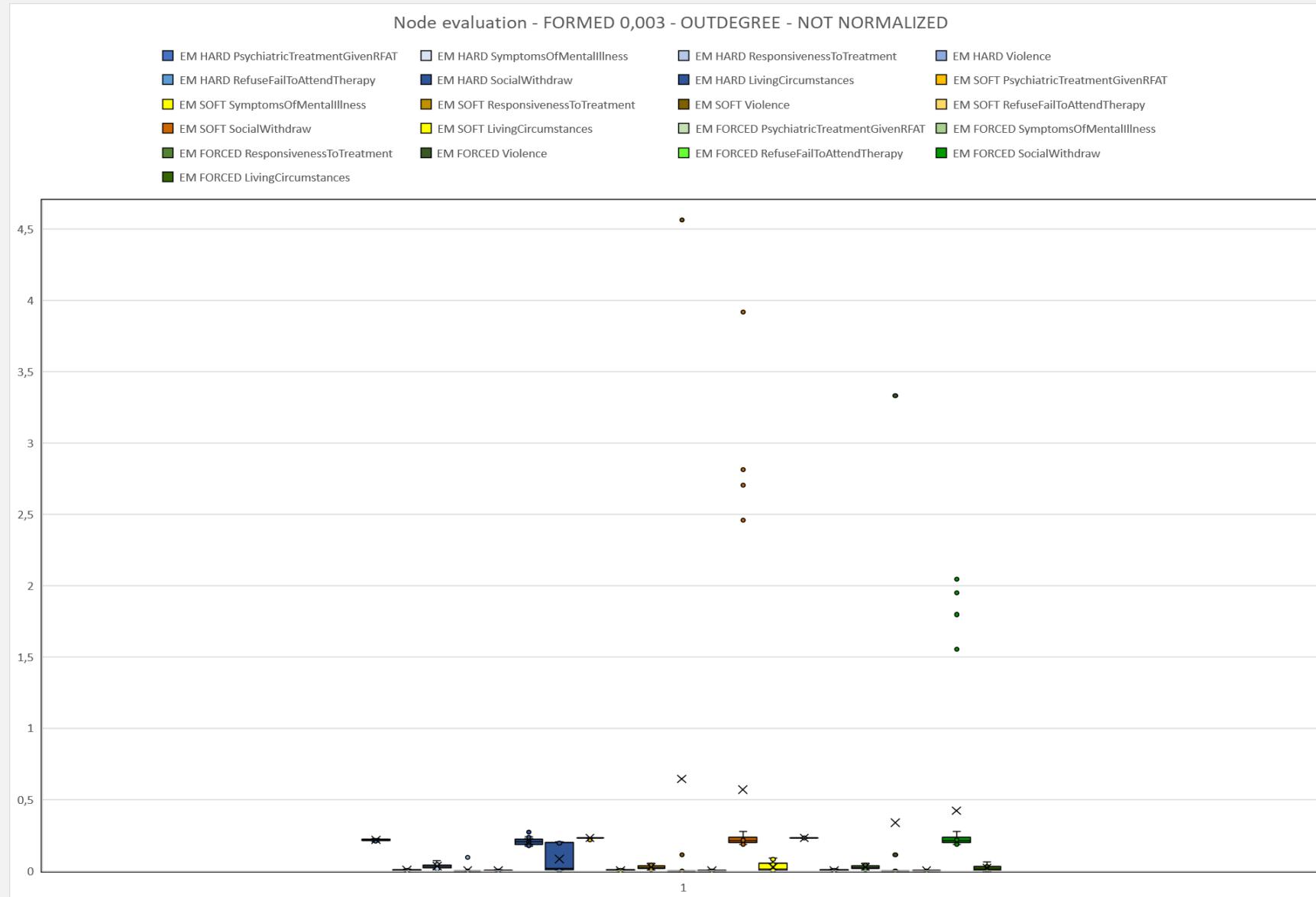


FORMED – GREATER OUTDEGREE – 0.003



# SPERIMENTAZIONI - FORMED

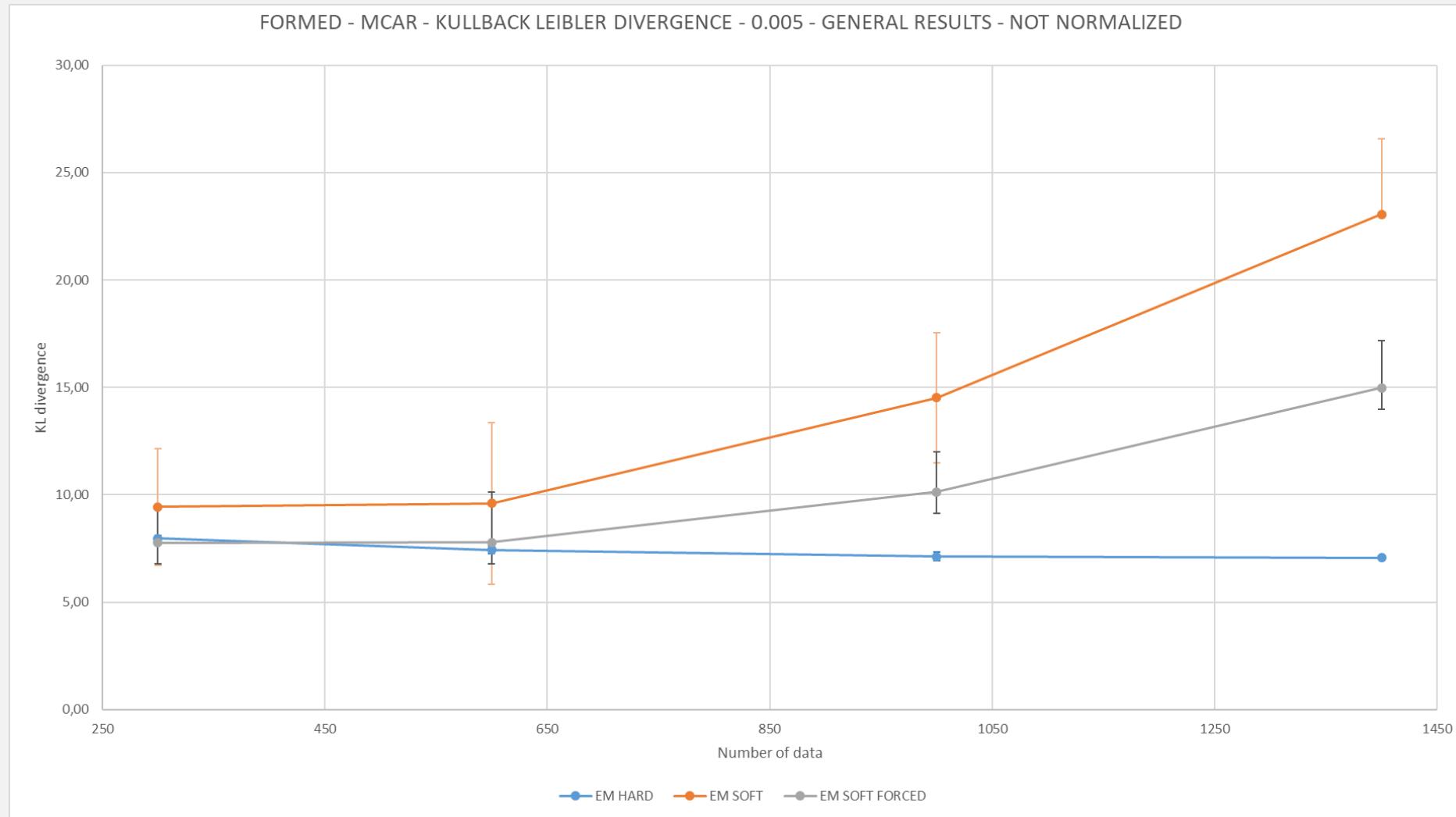
## FORMED – GREATER OUTDEGREE – 0.003



# SPERIMENTAZIONI - FORMED



FORMED – MCAR – 0.005

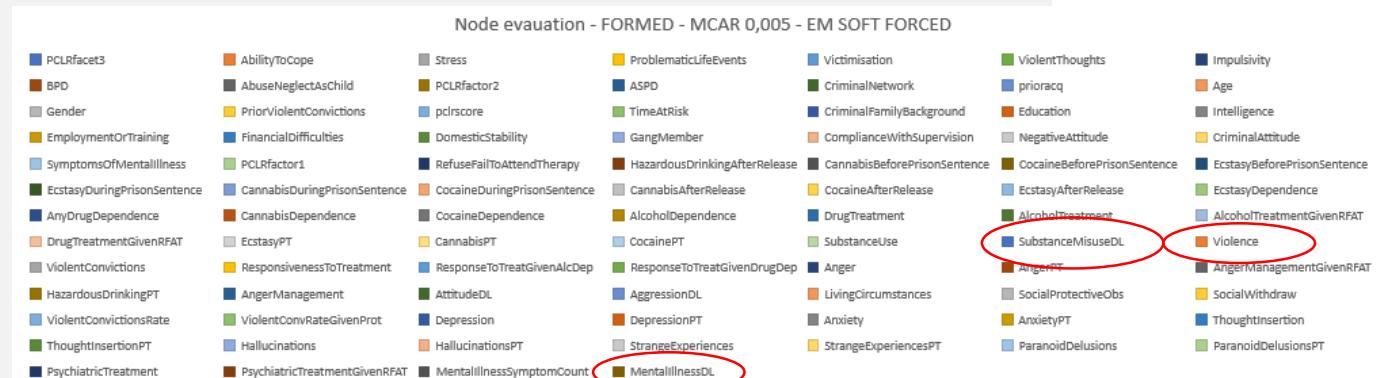
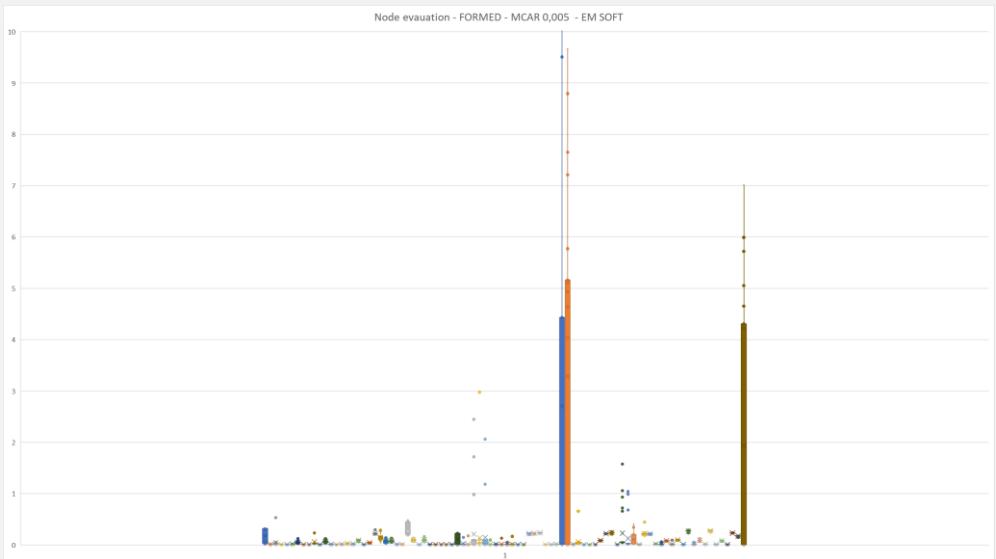
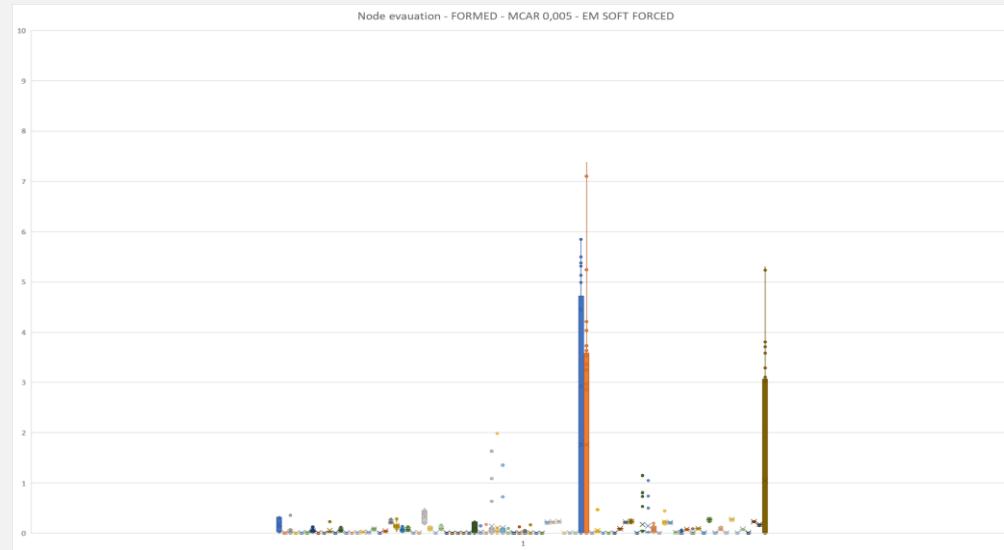


N.B Fenomeno di overfitting di EM SOFT che termina alla sesta iterazione

# SPERIMENTAZIONI - FORMED



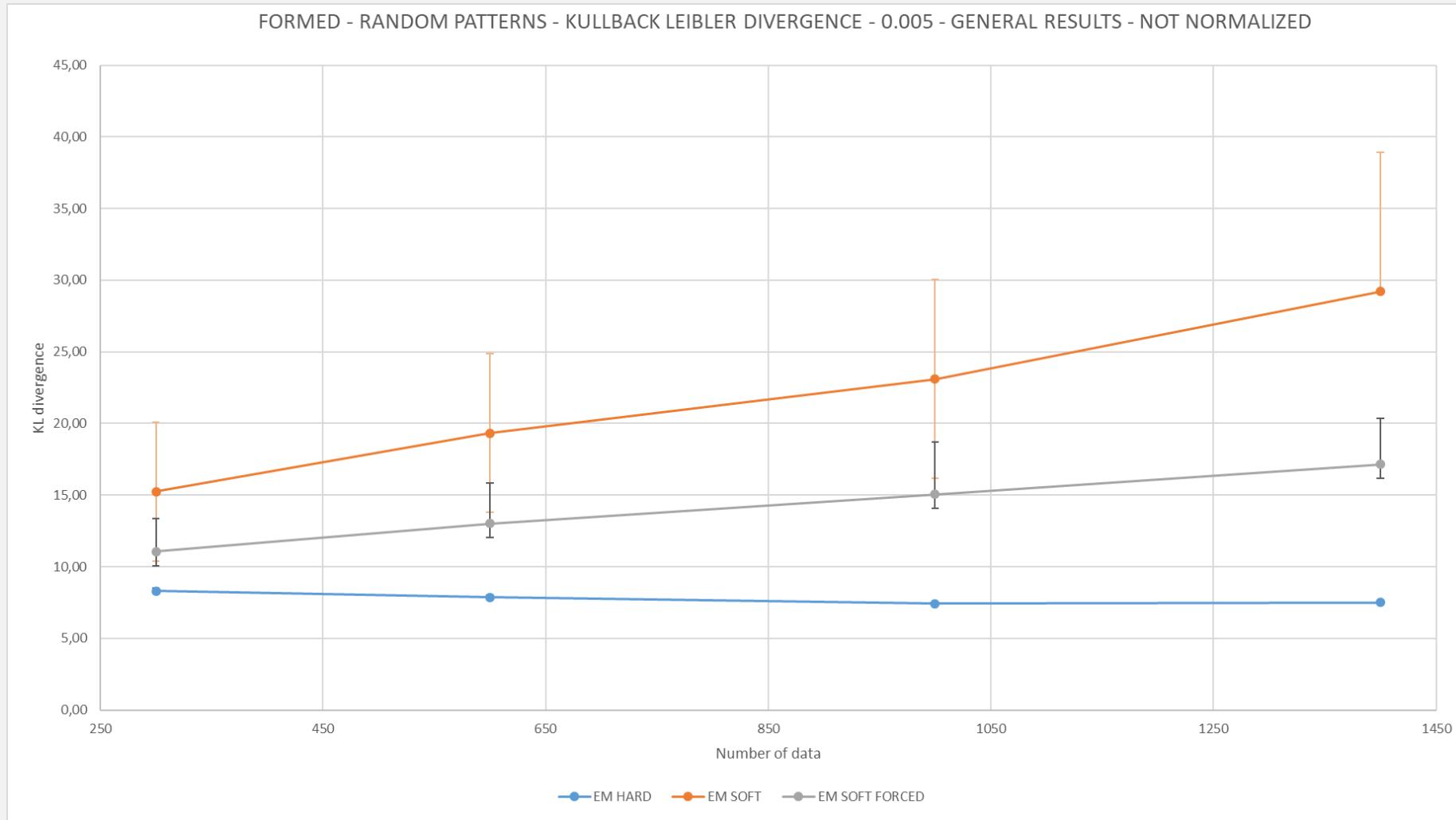
FORMED – MCAR – 0.005



# SPERIMENTAZIONI - FORMED



FORMED – RANDOM PATTERNS – 0.005

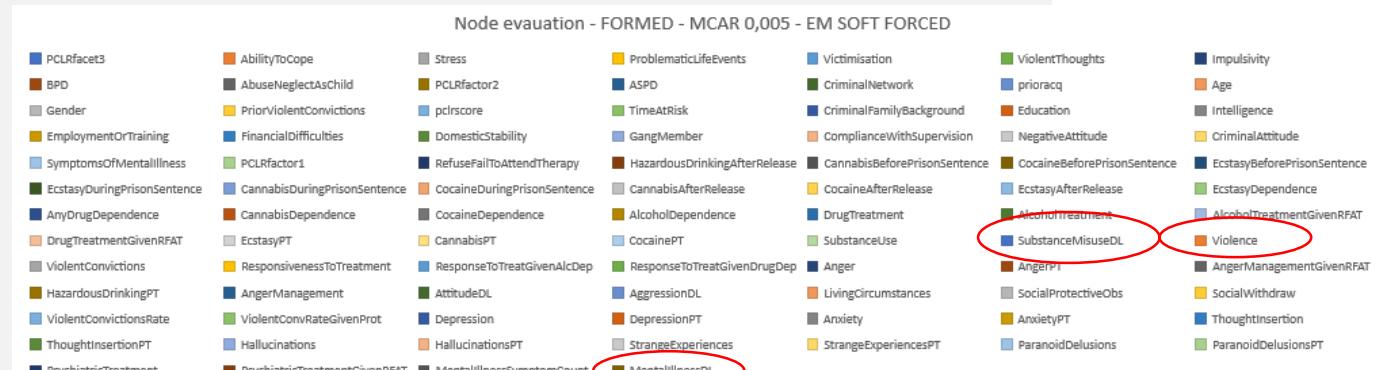
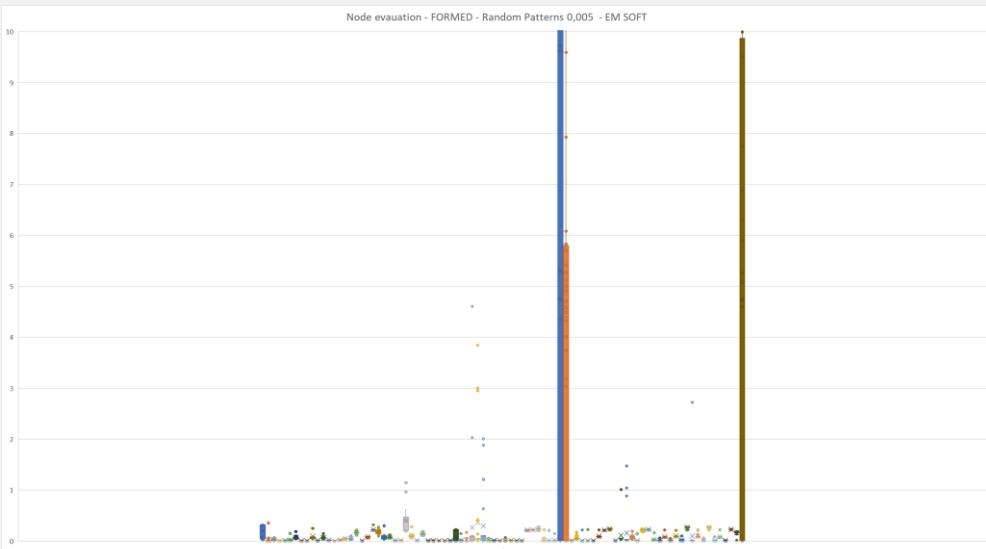
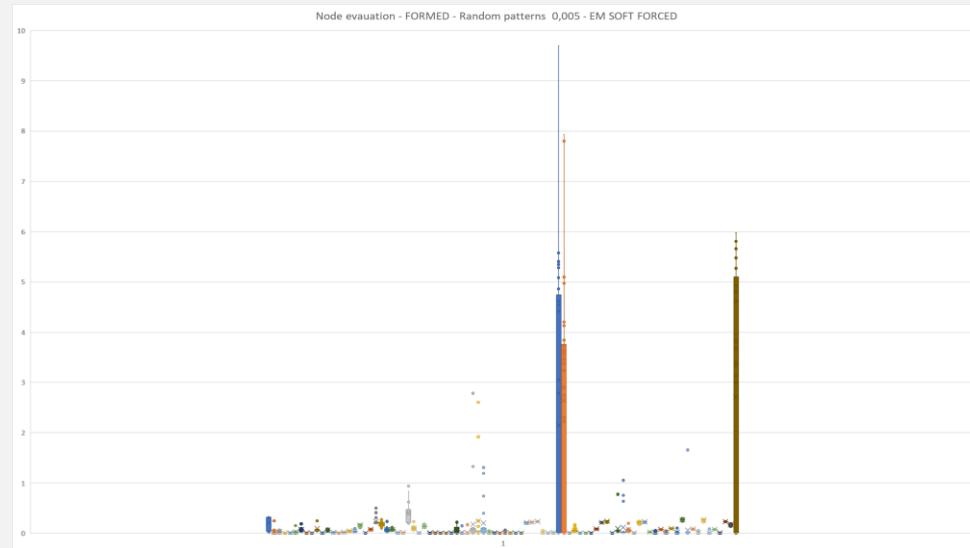


N.B Fenomeno di overfitting di EM SOFT che termina alla sesta iterazione

# SPERIMENTAZIONI - FORMED



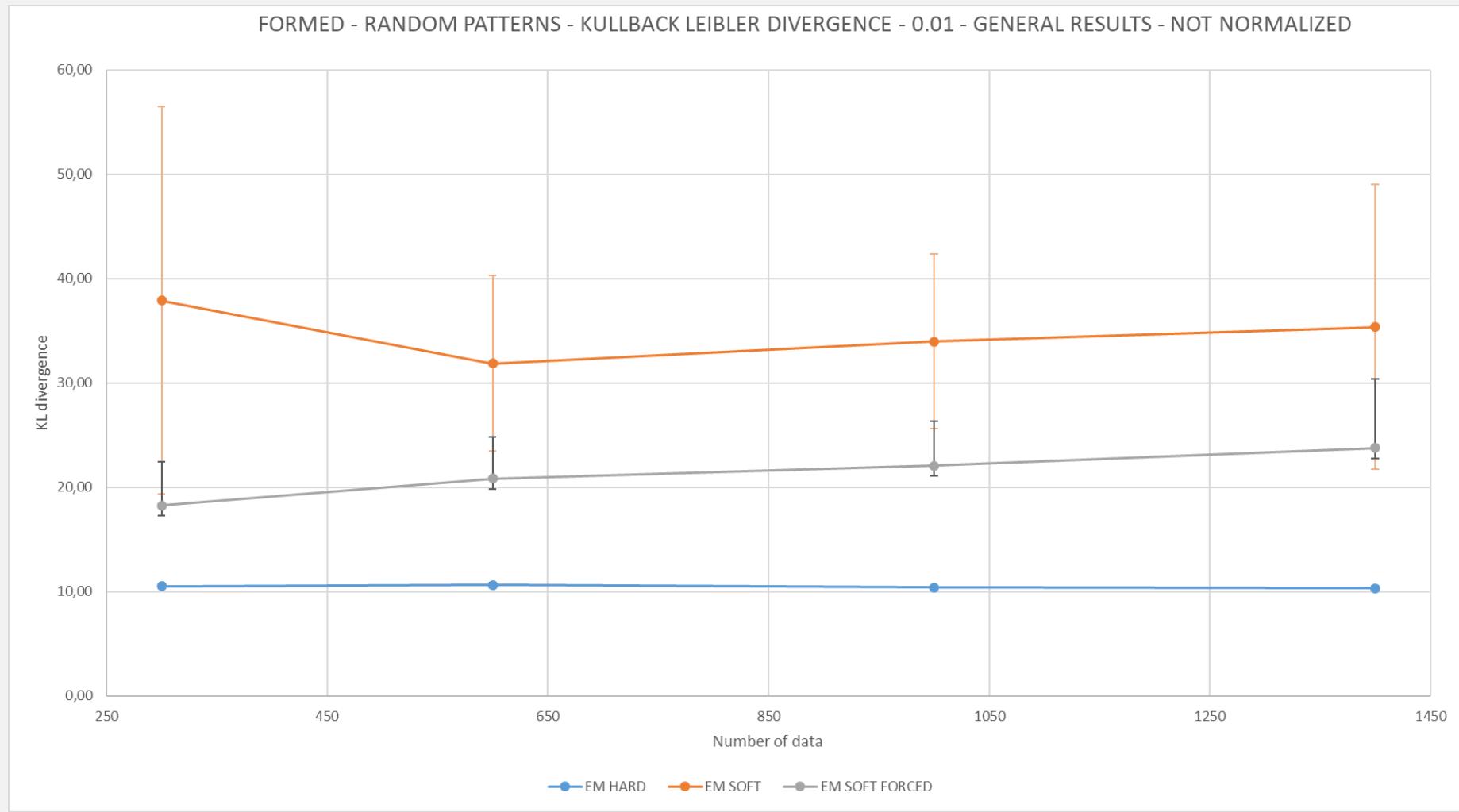
## FORMED – RANDOM PATTERNS – 0.005



# SPERIMENTAZIONI - FORMED



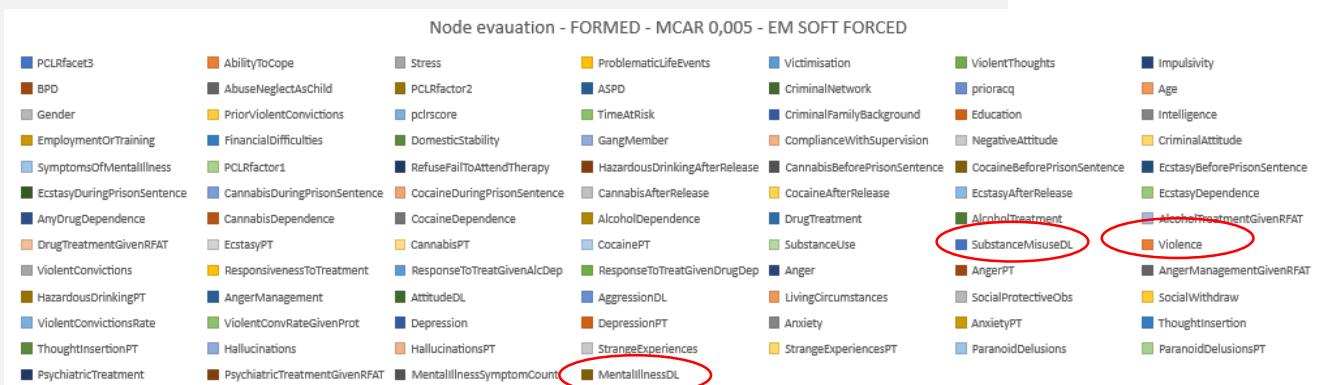
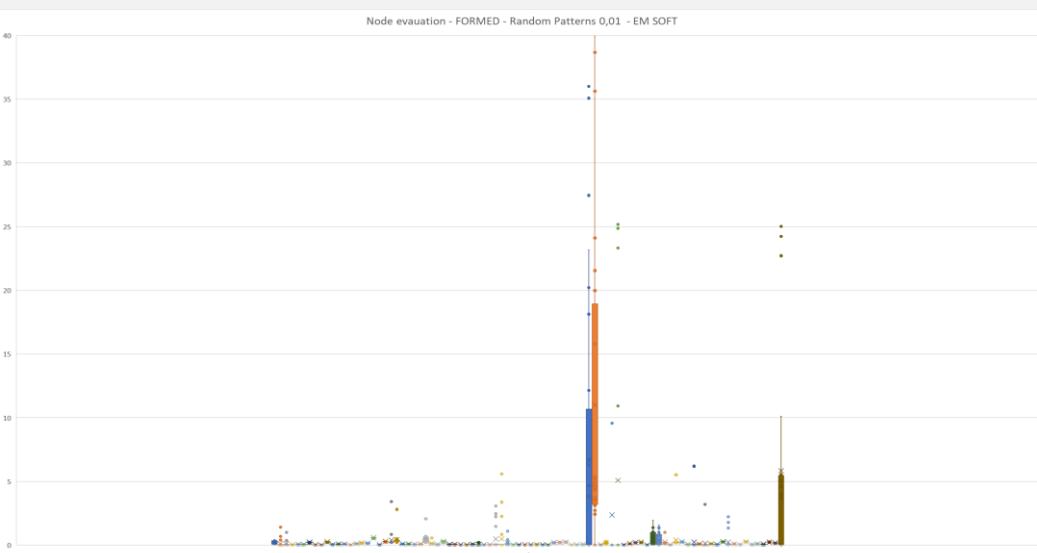
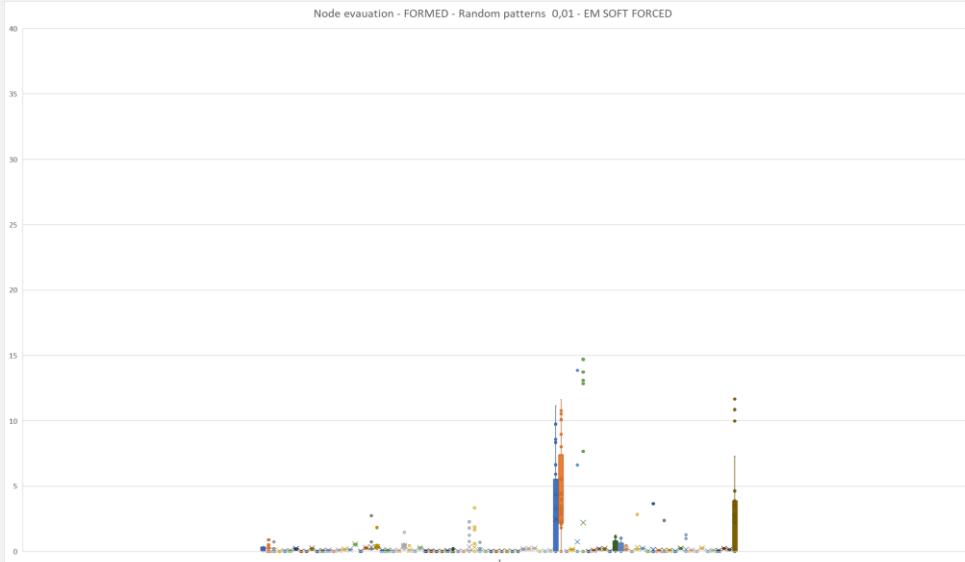
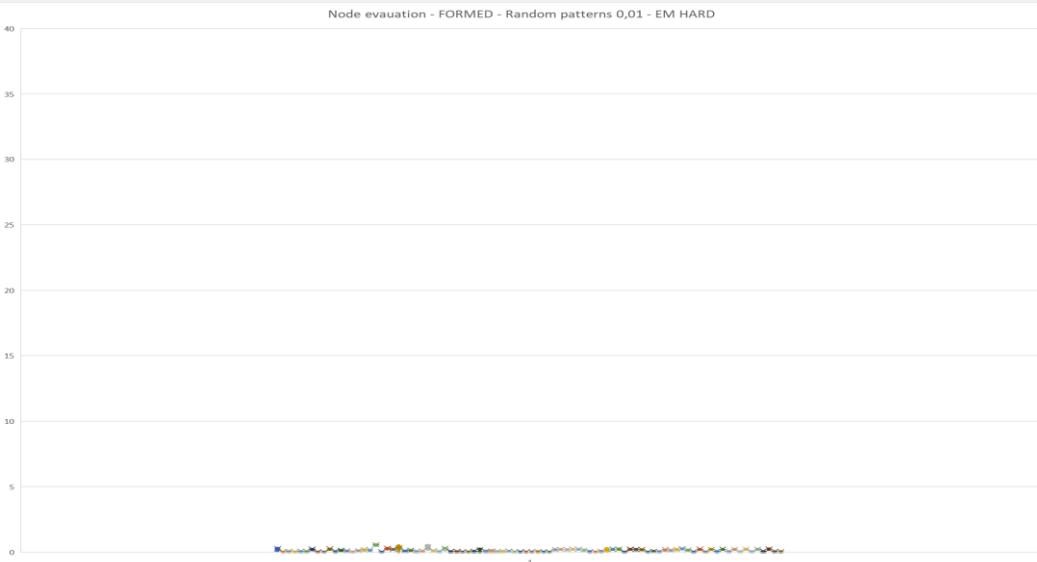
FORMED – RANDOM PATTERNS – 0.01



# SPERIMENTAZIONI - FORMED

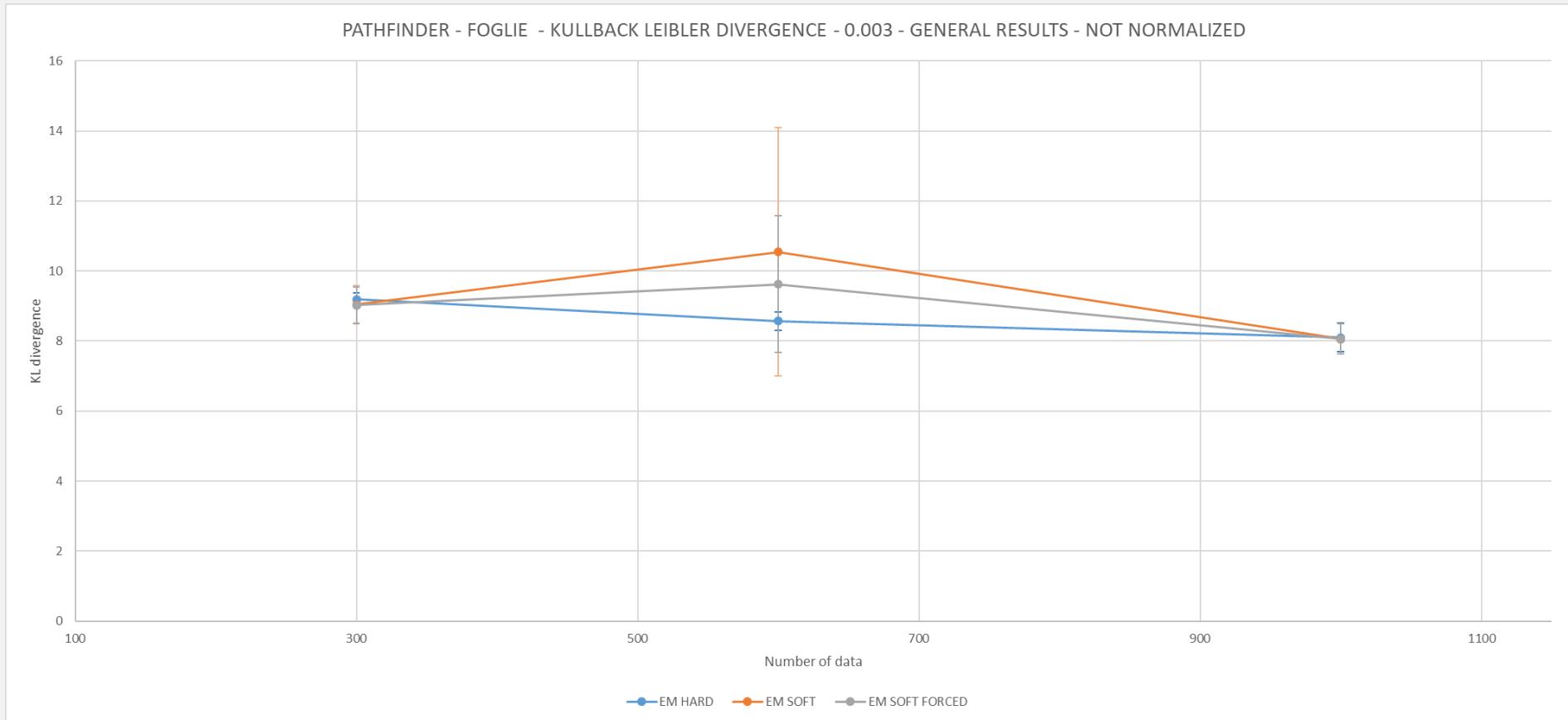


## FORMED – RANDOM PATTERNS – 0.01



# SPERIMENTAZIONI - PATHFINDER

## PATHFINDER – FOGLIE – 0.003



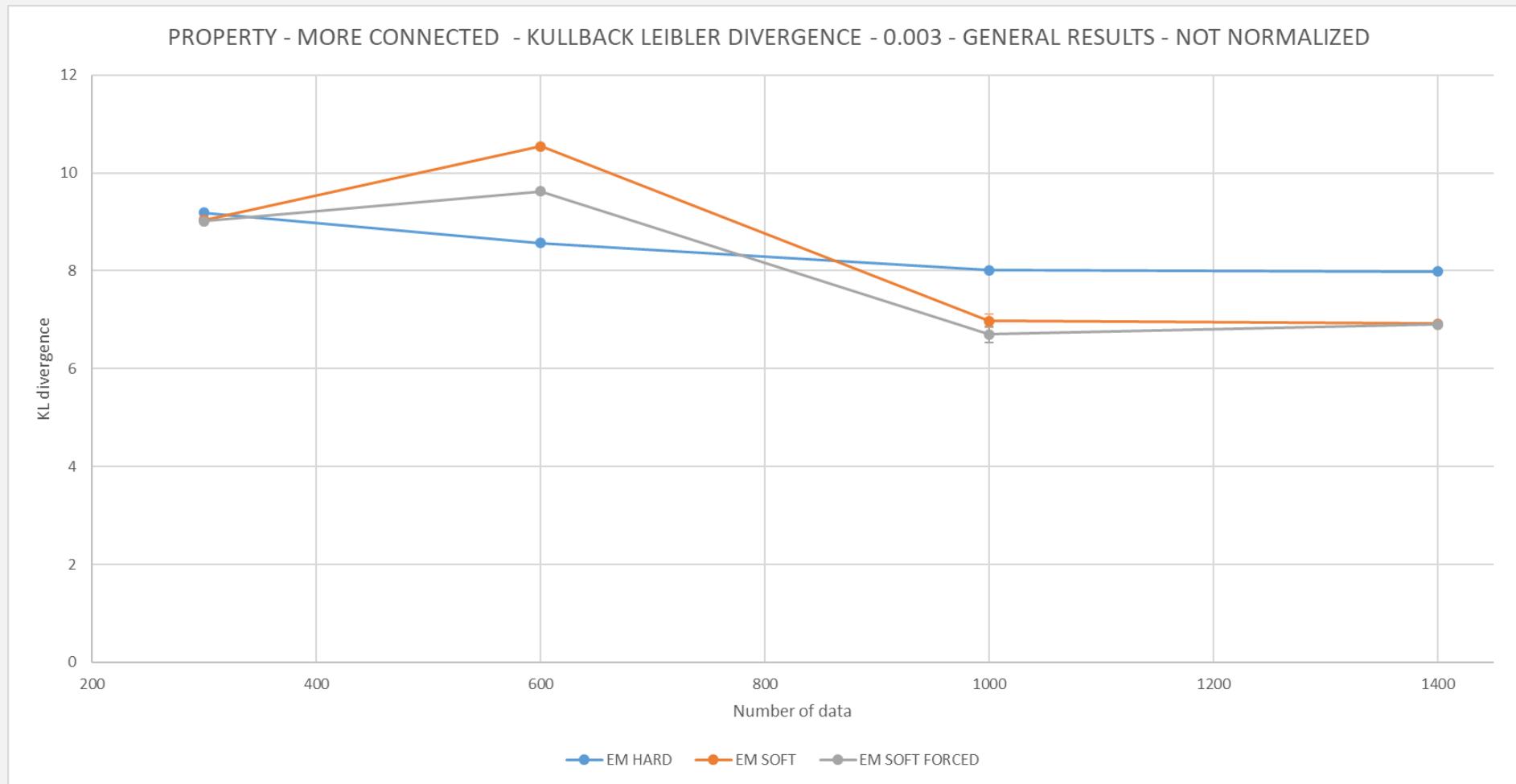
# SPERIMENTAZIONI - PATHFINDER

## PATHFINDER – FOGLIE – 0.003

Dato l'alto **numero di nodi foglia** all'interno della rete *Pathfinder* ( $> 20$ ), è stato deciso di non riportare il grafico comparativo come nei casi precedenti. Tuttavia, effettuando un confronto numerico tra i valori ottenuti, **non vengono segnalati differenze significative tra i singoli nodi**

# Sperimentazioni - PATHFINDER

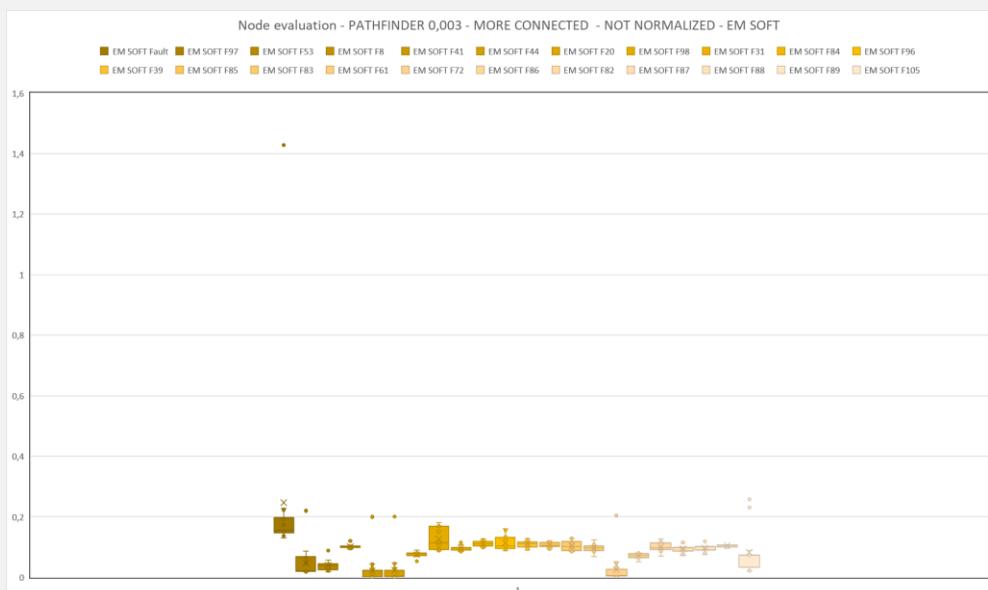
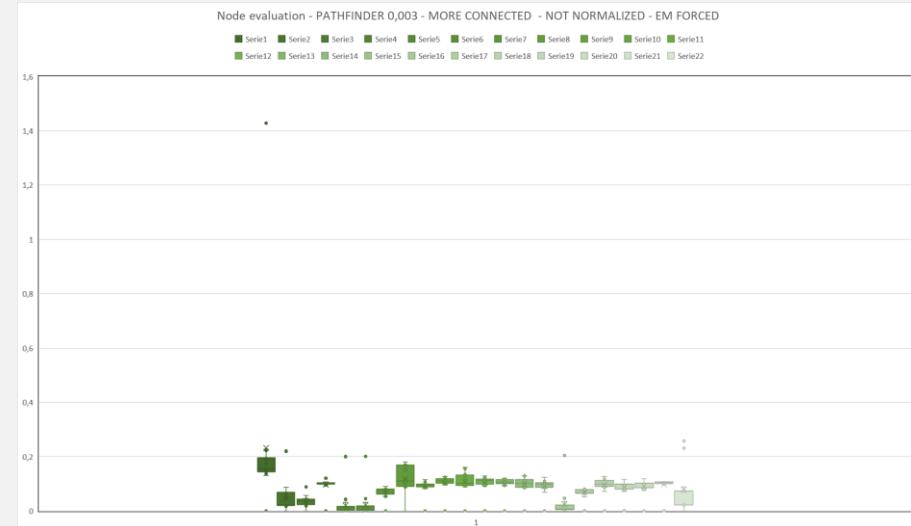
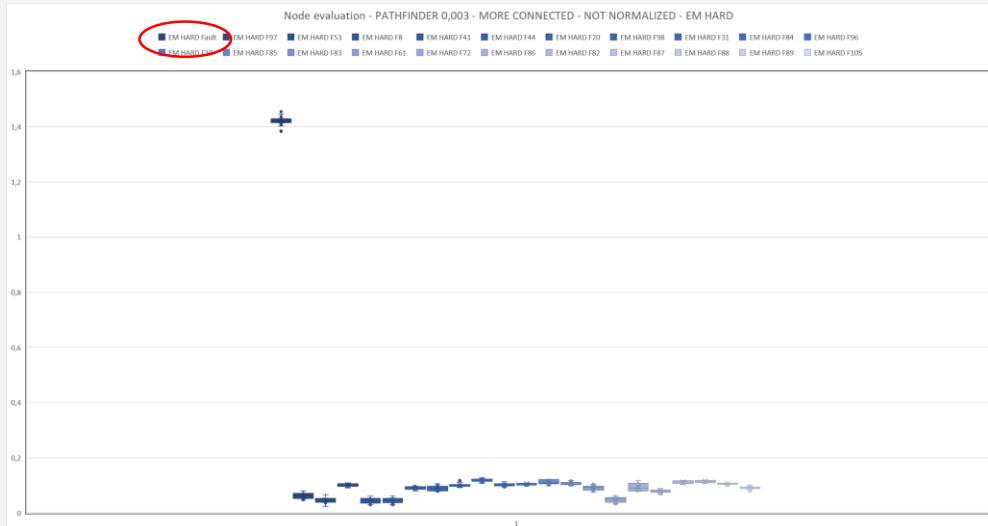
## PATHFINDER – MORE CONNECTED – 0.003



# SPERIMENTAZIONI – PATHFINDER



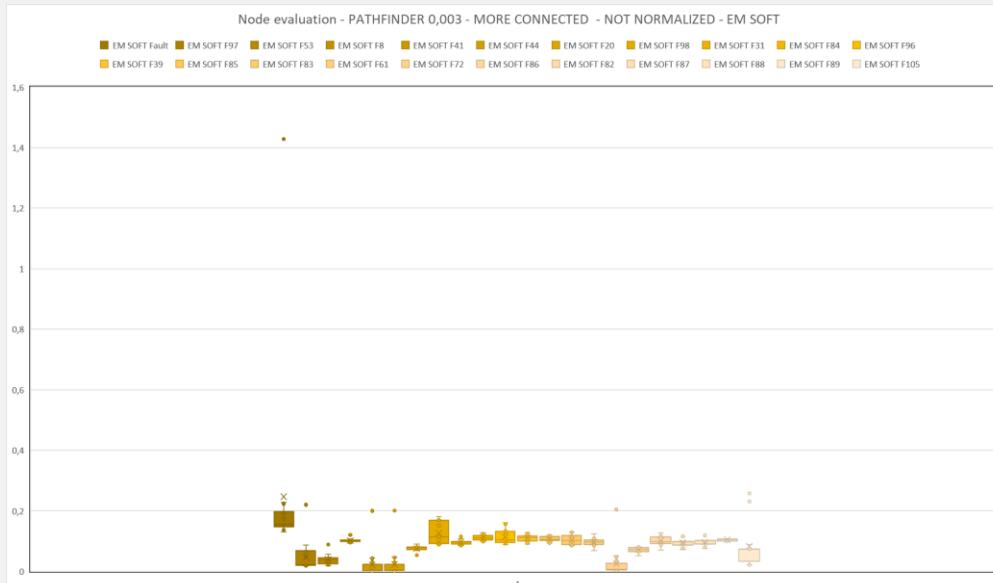
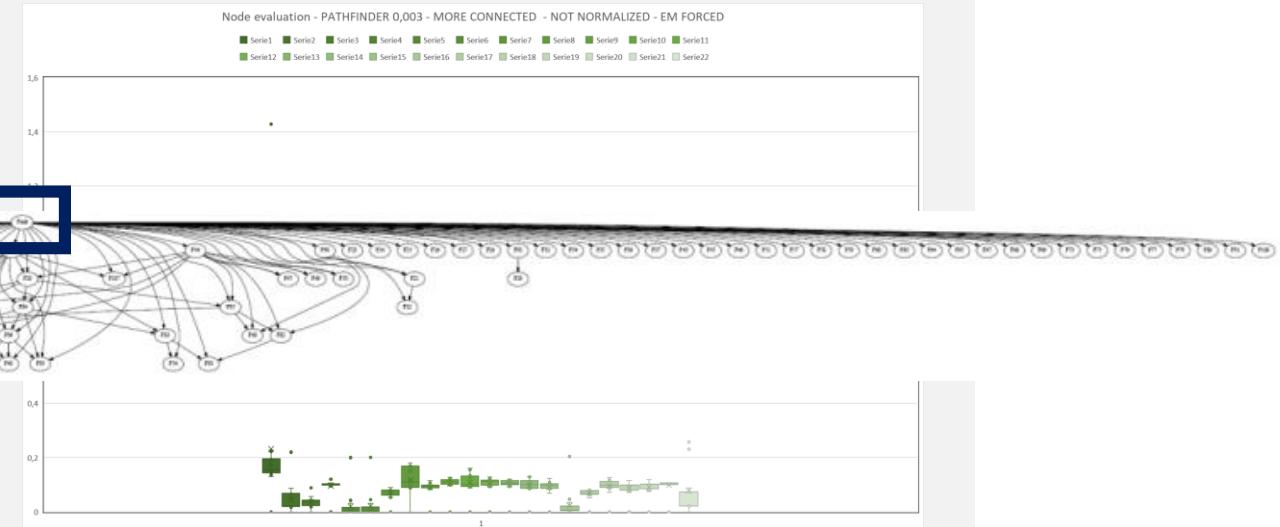
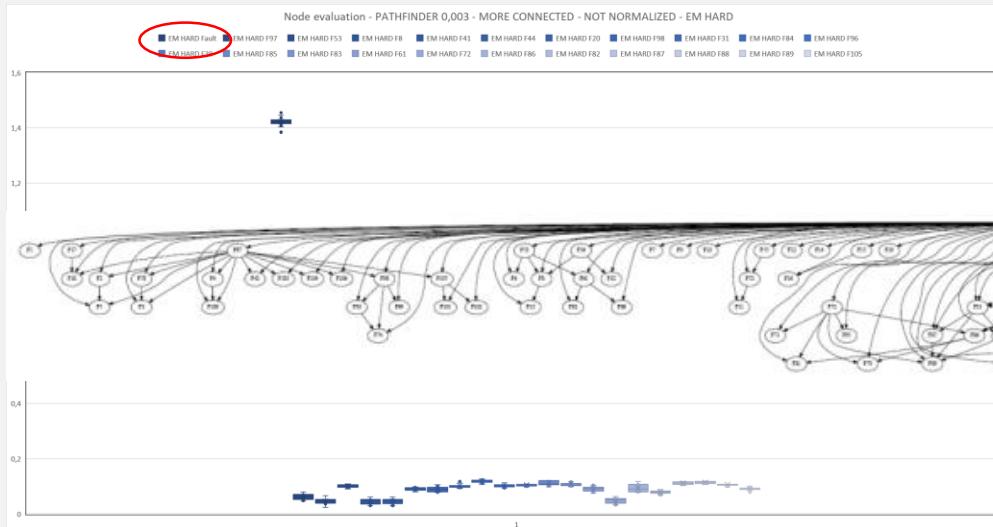
## PATHFINDEER – MORE CONNECTED – 0.003



# SPERIMENTAZIONI – PATHFINDER

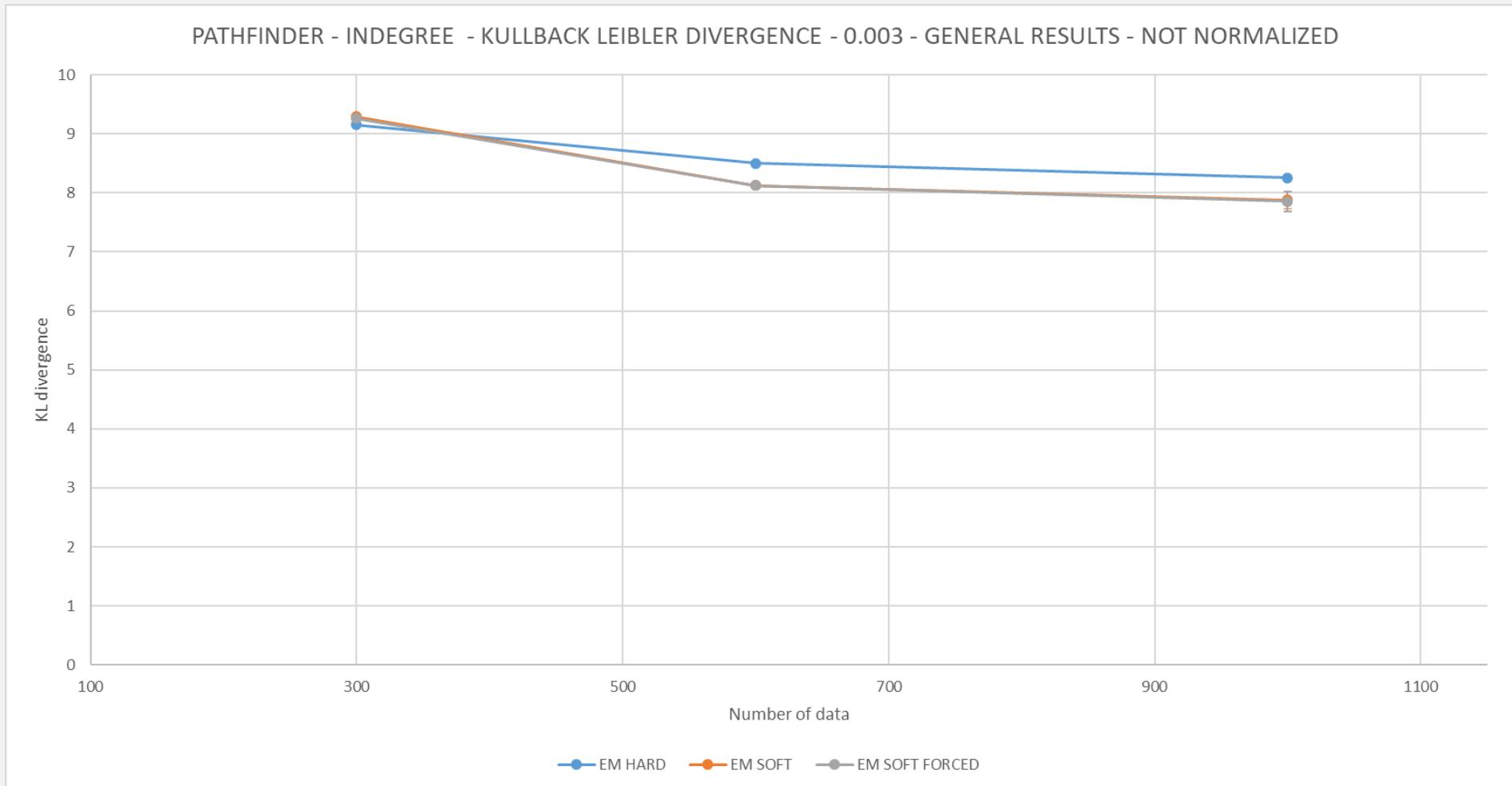


## PATHFINDEER – MORE CONNECTED – 0.003



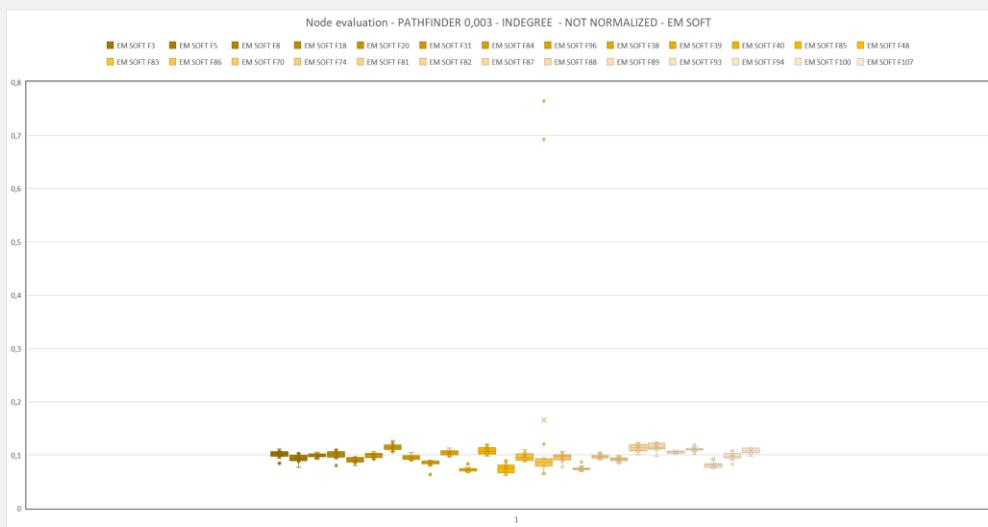
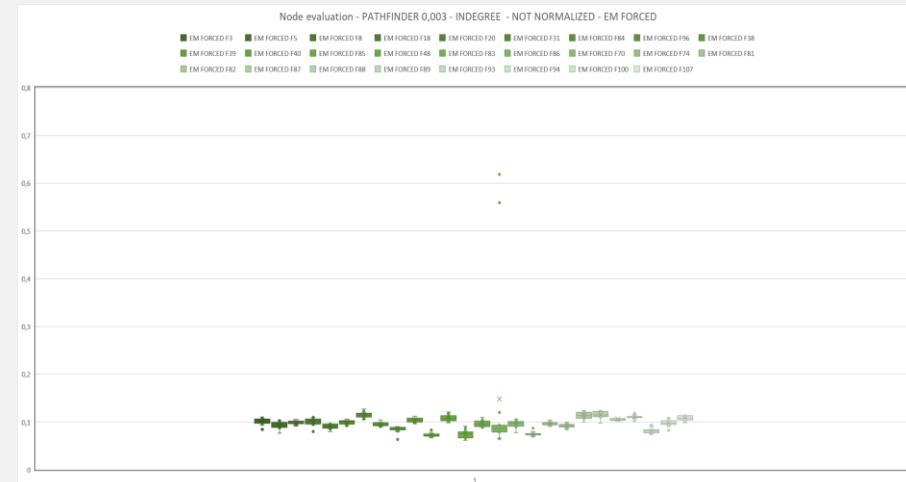
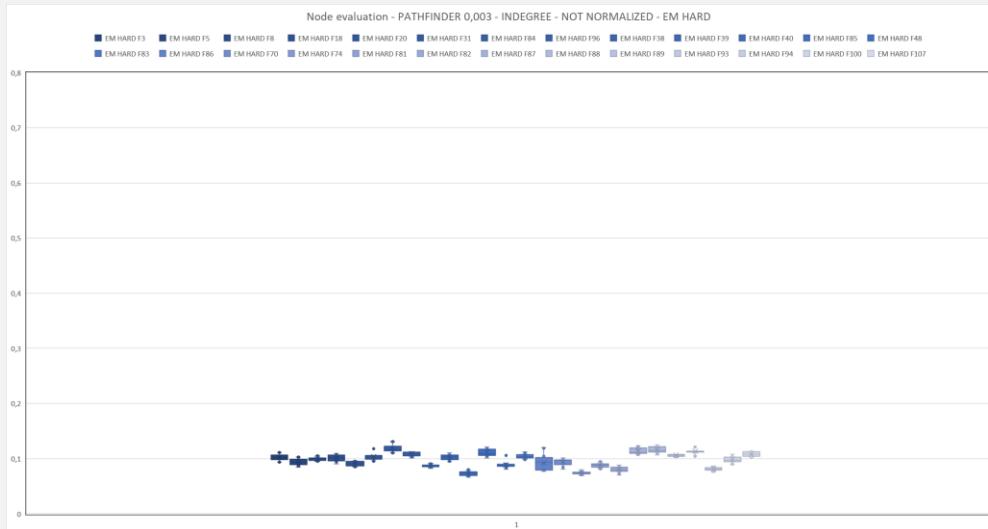
# SPERIMENTAZIONI - PATHFINDER

PATHFINDER – INDEGREE – 0.003



# SPERIMENTAZIONI – PATHFINDER

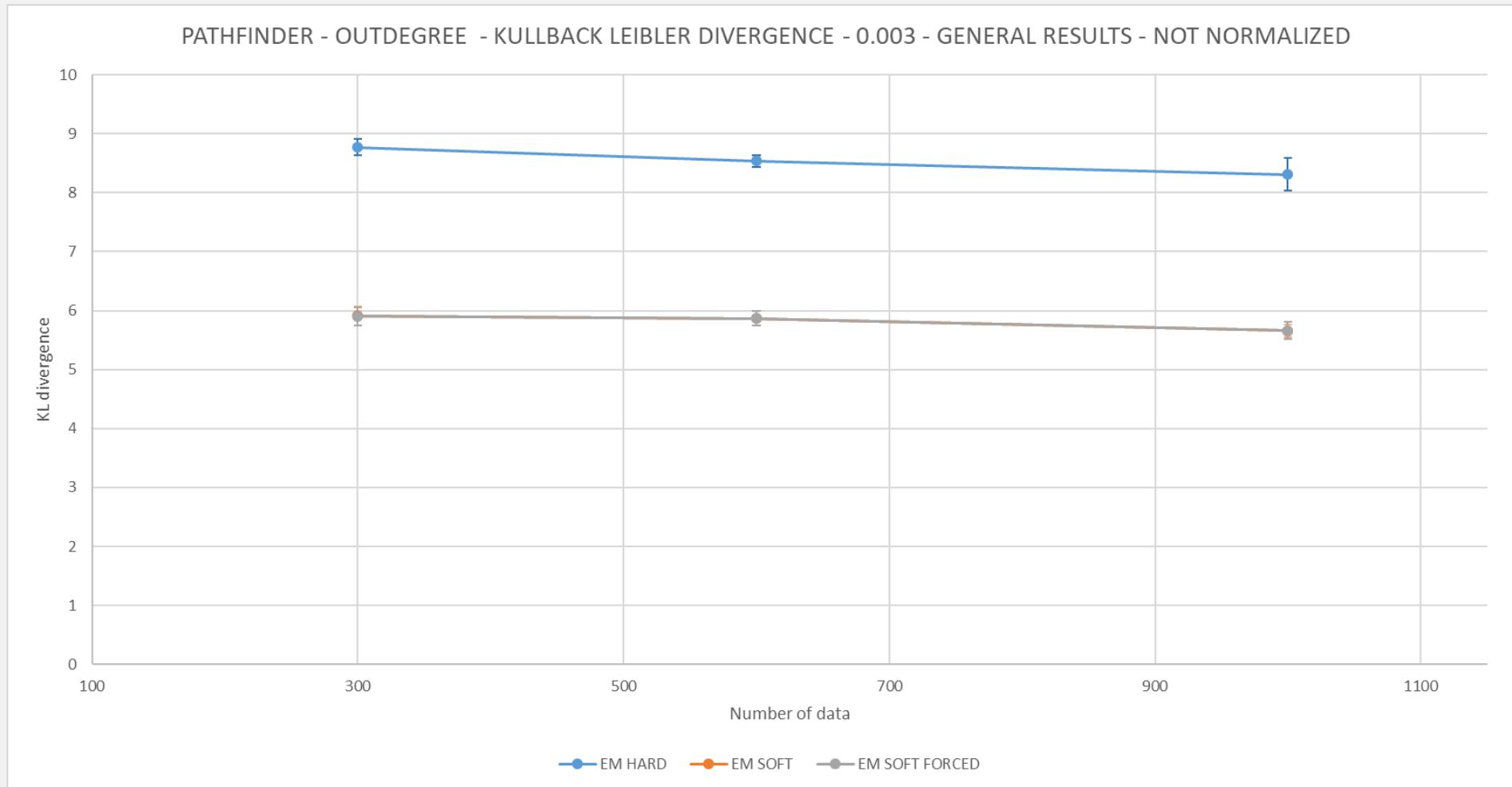
## PATHFINDEER – INDEGREE – 0.003



# SPERIMENTAZIONI - PATHFINDER



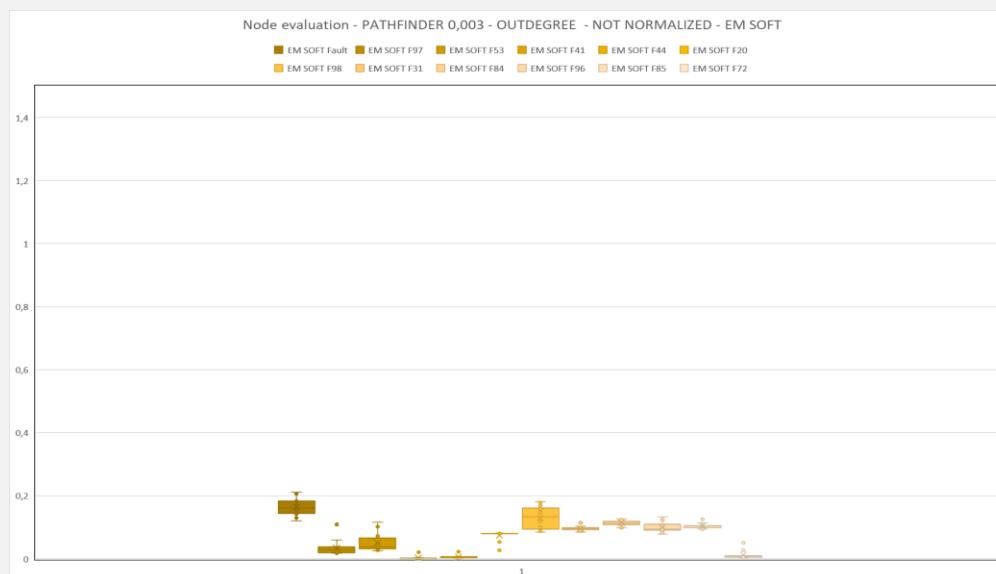
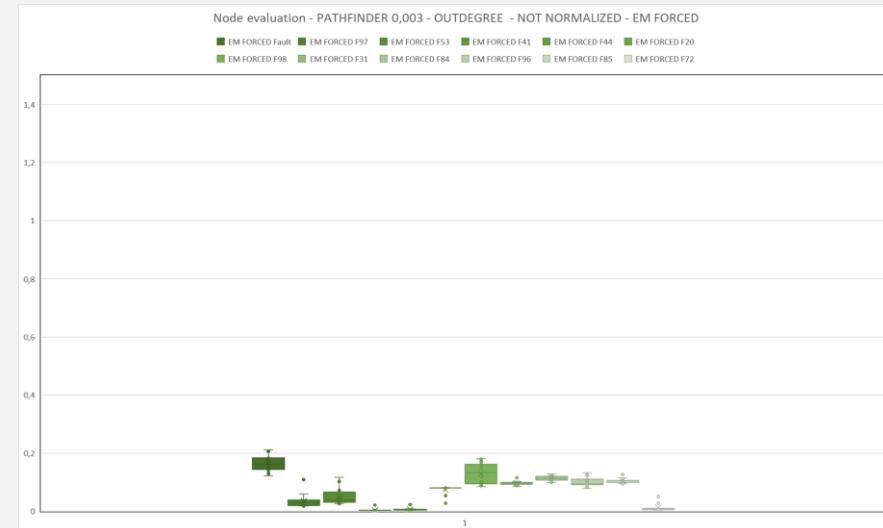
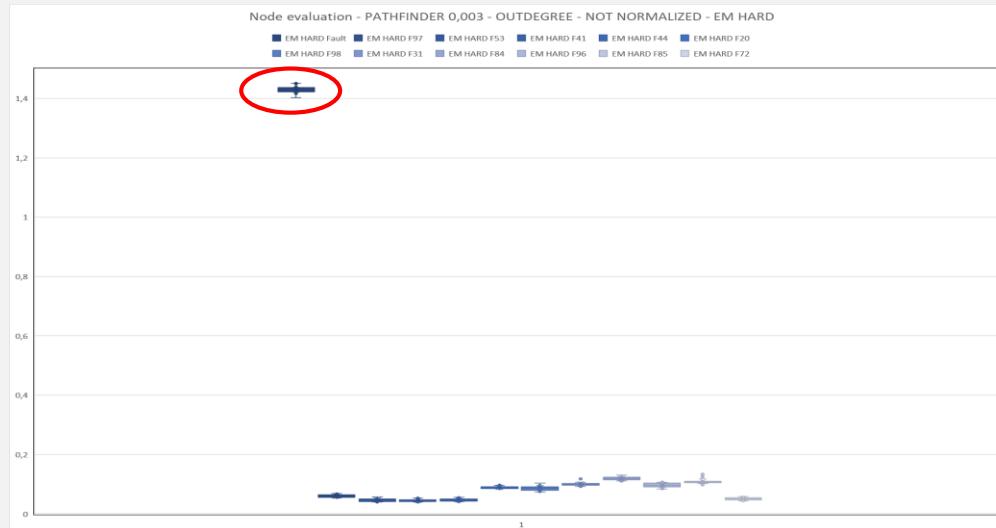
PATHFINDER – OUTDEGREE – 0.003



# SPERIMENTAZIONI – PATHFINDER



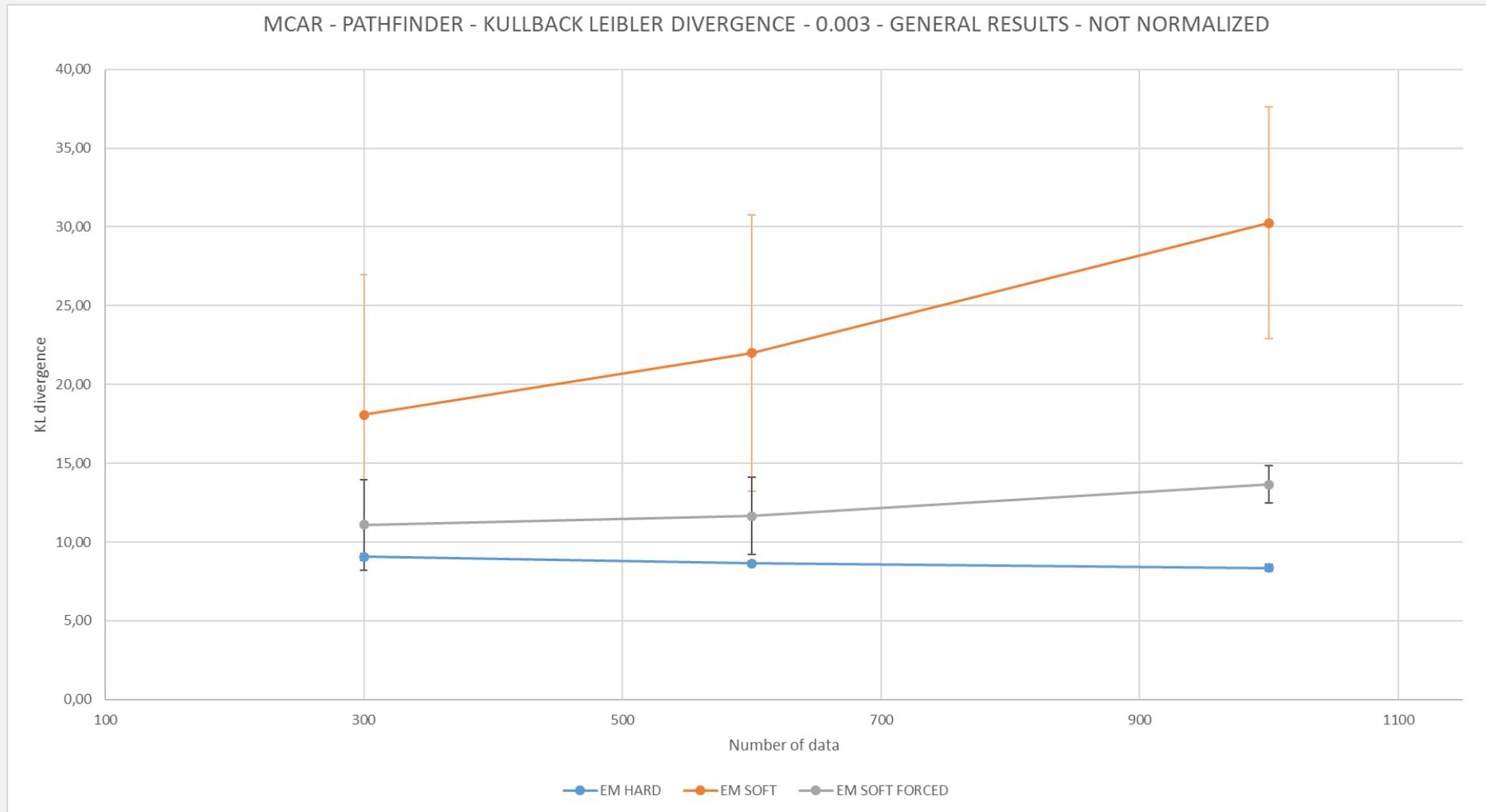
PATHFINDEER – OUTDEGREE – 0.003



# SPERIMENTAZIONI - PATHFINDER



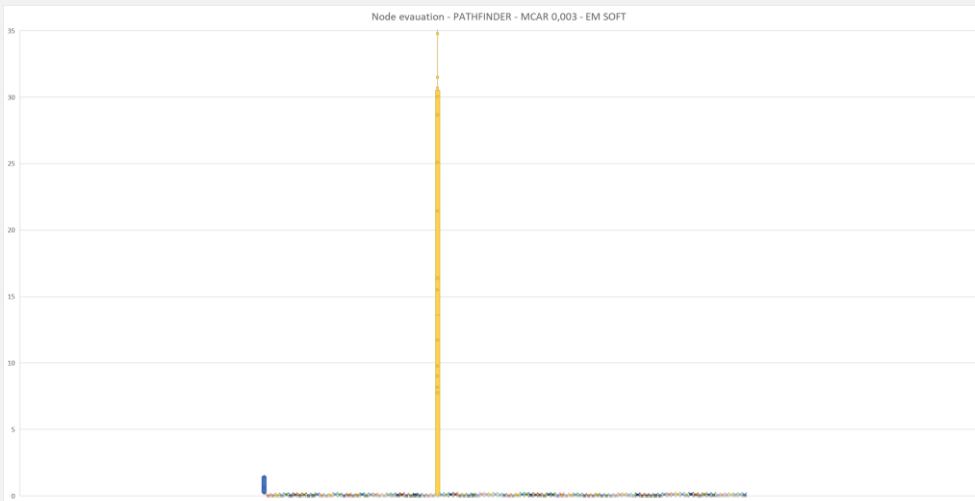
PATHFINDER – MCAR – 0.005



# SPERIMENTAZIONI - PATHFINDER



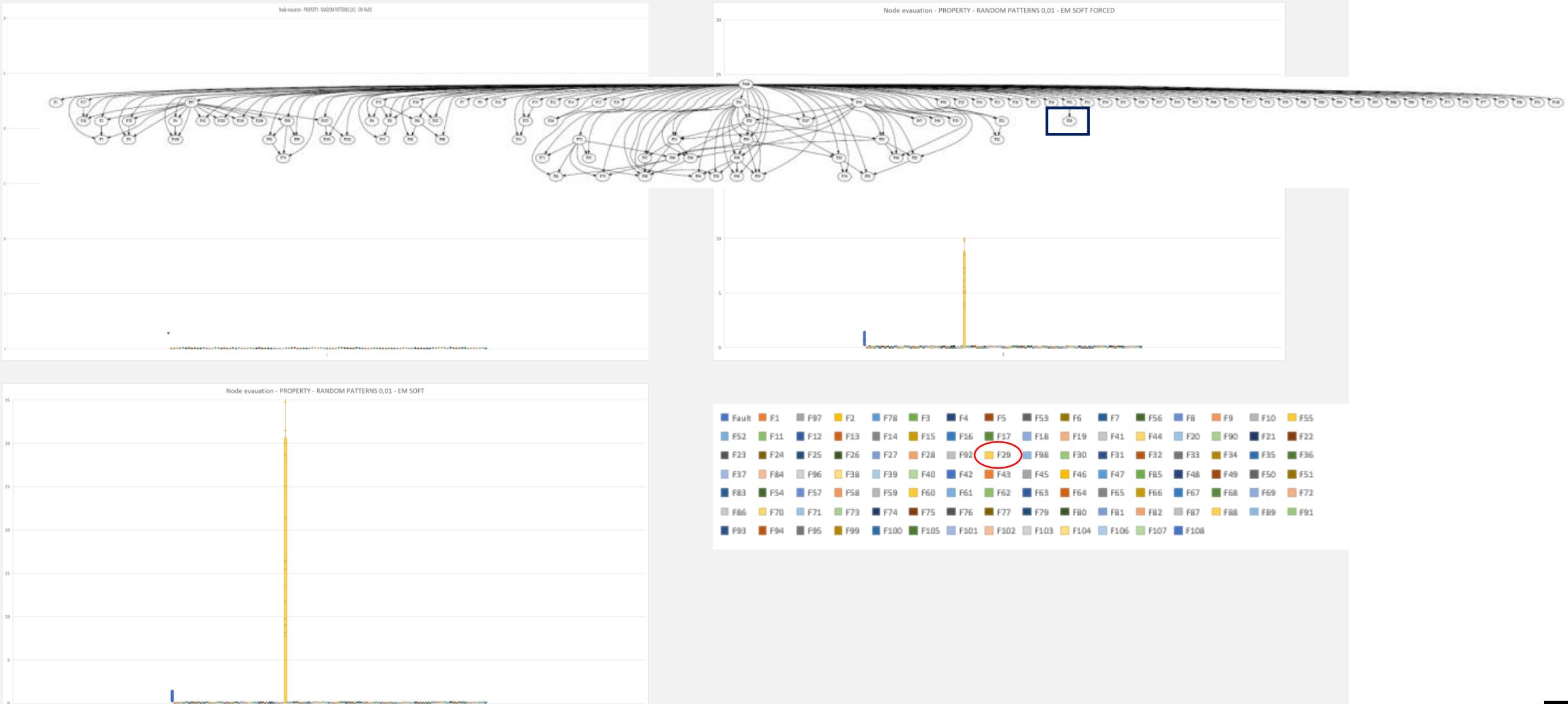
PATHFINDER – MCAR – 0.005



# SPERIMENTAZIONI - PATHFINDER



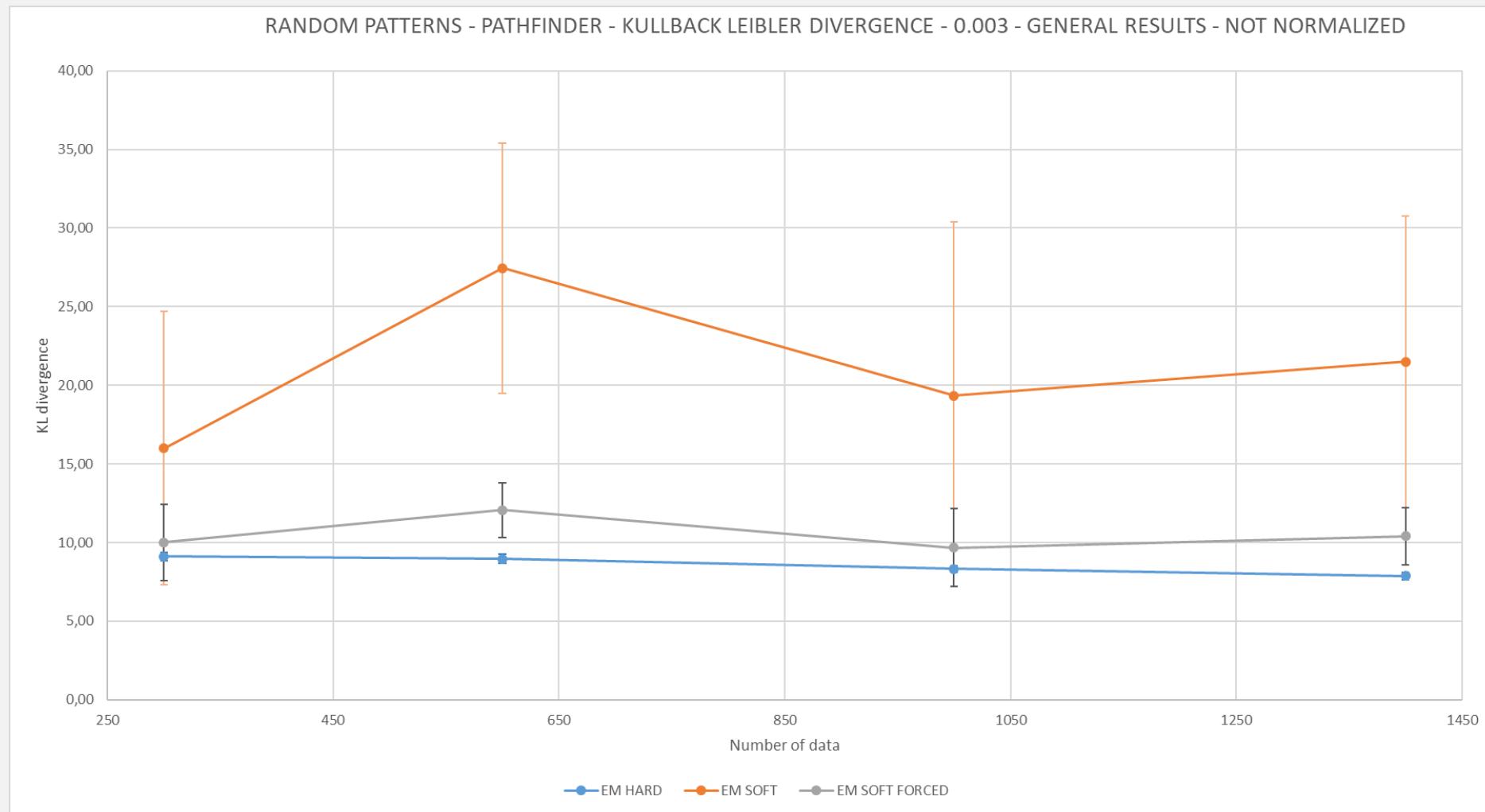
PATHFINDER – MCAR – 0.005



# SPERIMENTAZIONI - PATHFINDER

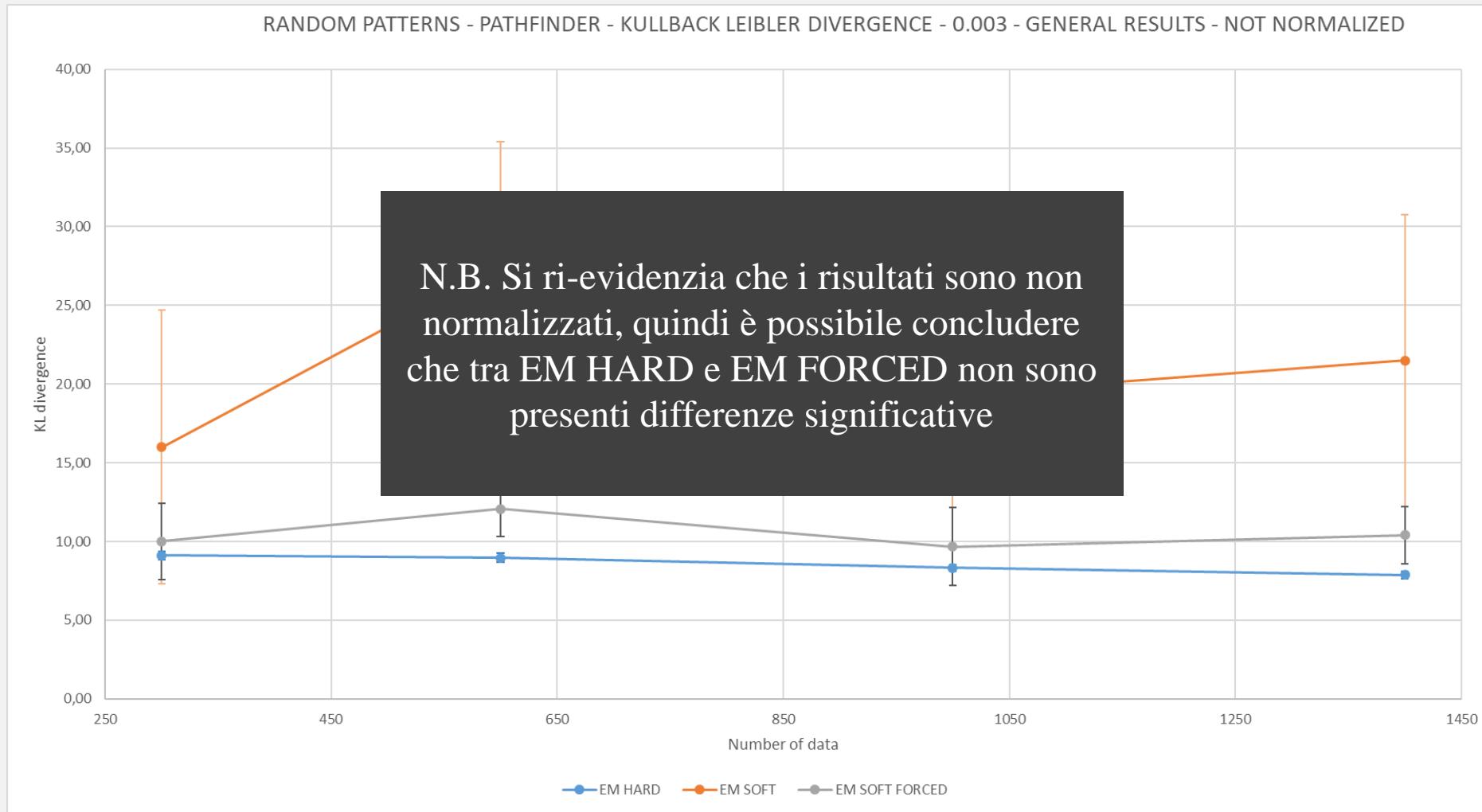


## PATHFINDER – RANDOM PATTERNS – 0.003



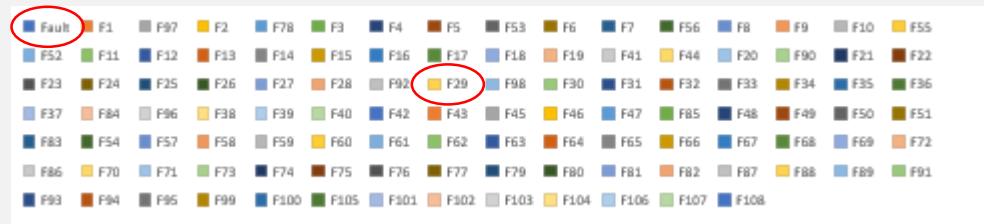
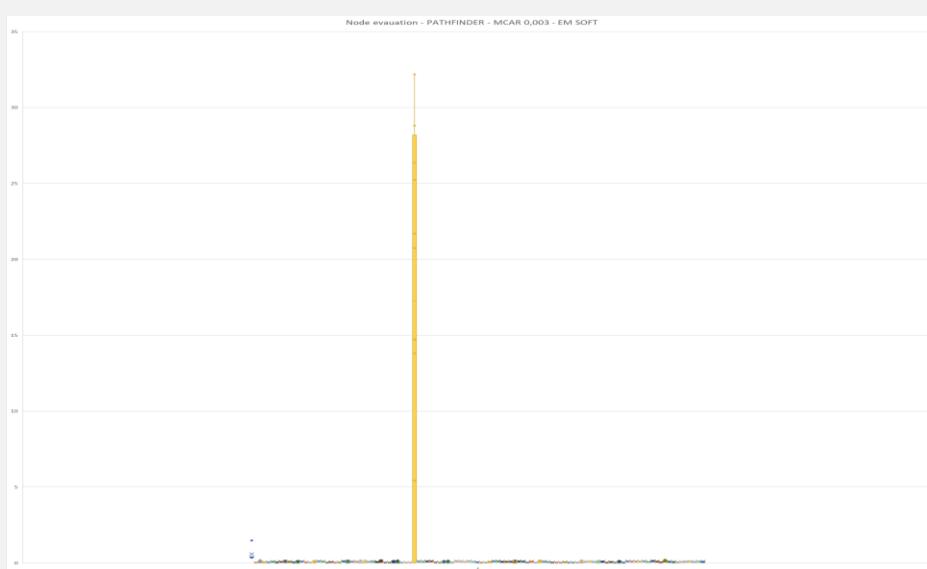
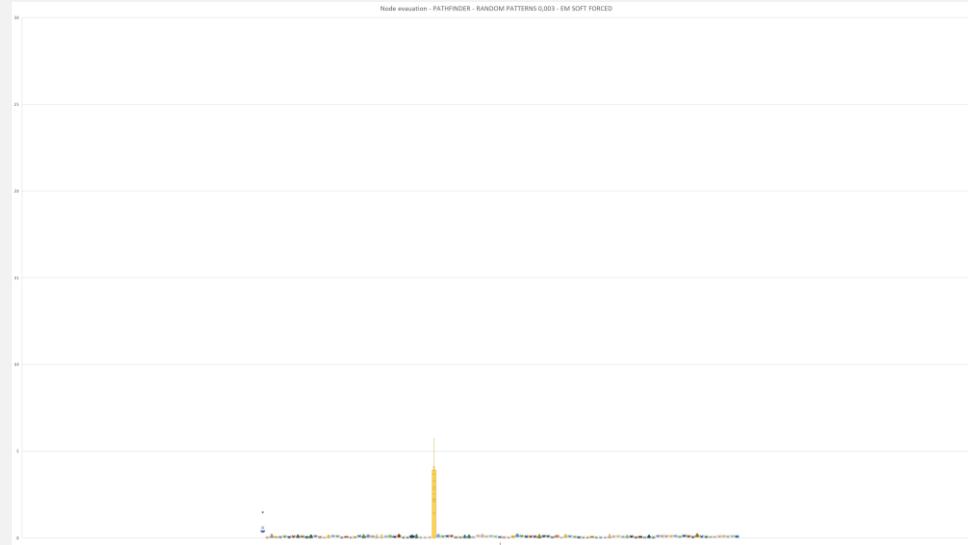
# SPERIMENTAZIONI - PATHFINDER

## PATHFINDER – RANDOM PATTERNS – 0.003



# SPERIMENTAZIONI - PATHFINDER

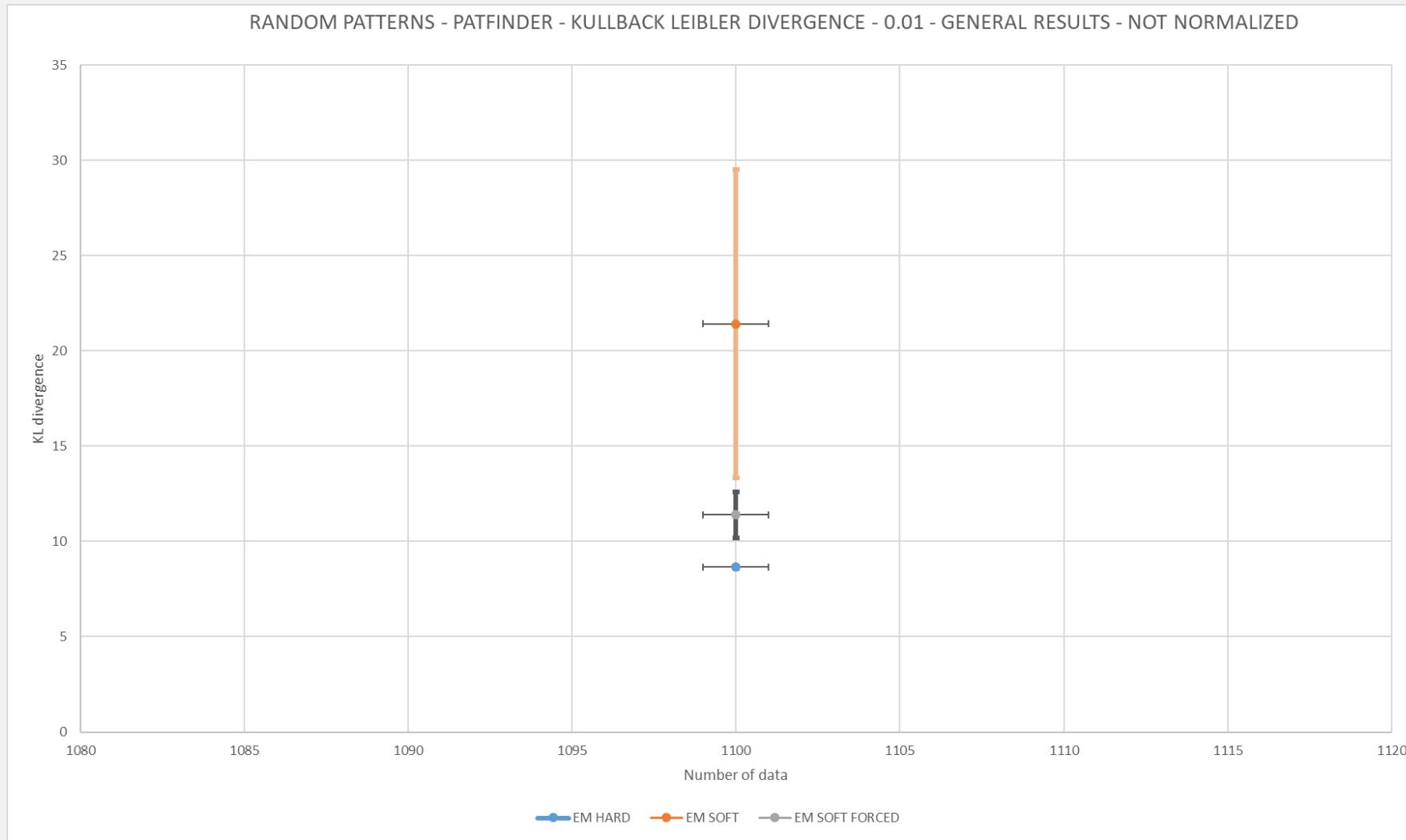
## PATHFINDER – RANDOM PATTERNS – 0.003



EM HARD fitta meglio la variabile F29.  
Tuttavia, fitta peggio altre variabili come Fault

# Sperimentazioni - PATHFINDER

## PATHFINDER – RANDOM PATTERNS – 0.01



# SPERIMENTAZIONI - PATHFINDER

## *PATHFINDER – RANDOM PATTERNS – 0.01*

Per quanto riguarda, i singoli nodi, valgono le stesse considerazioni fatte alla slide 79

# Tempo computazionale

*Sports*  
*Property*  
*ForMed*  
*PathFinder*

# ANALISI DEL TEMPO COMPUTAZIONALE

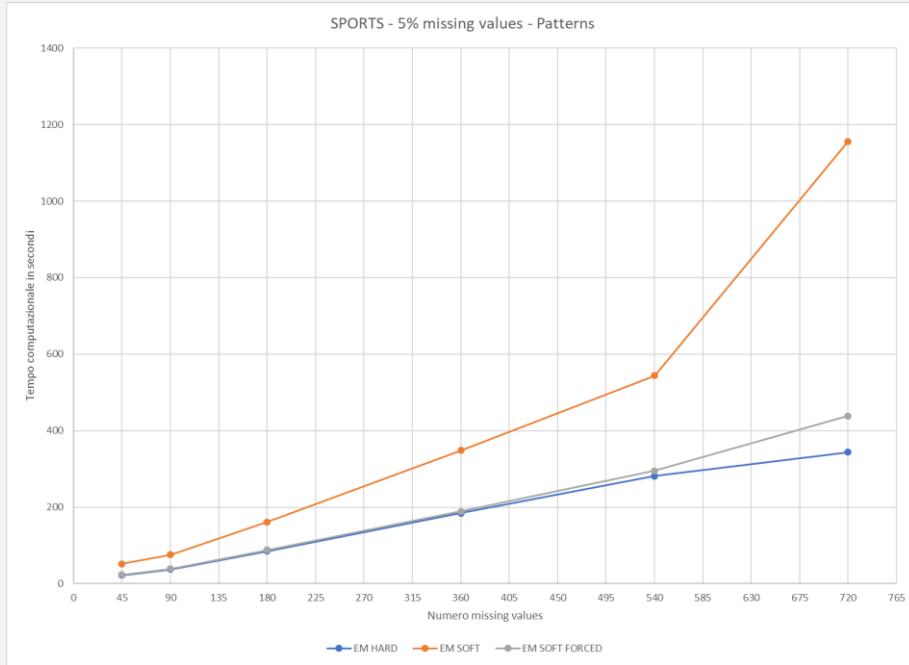
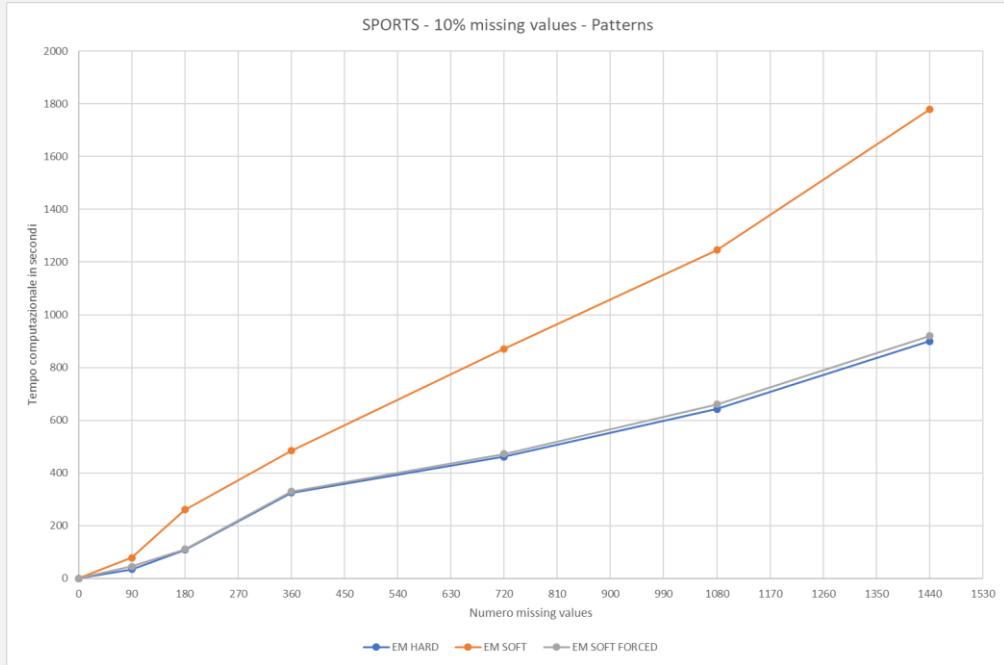
## Considerazioni:

- Tutti gli esperimenti sono stati eseguiti nella stessa macchina, **senza parallelizzazione** del codice sorgente.
  - Tutti gli esperimenti sono stati eseguiti con il **codice ottimizzato**.
- Per ogni iterazione, sono state eseguite **5 repliche** e mediati i risultati. Non sono stati riportati gli intervalli di confidenza in quanto gli scostamenti rispetto alla media risultano essere **non significativi**.

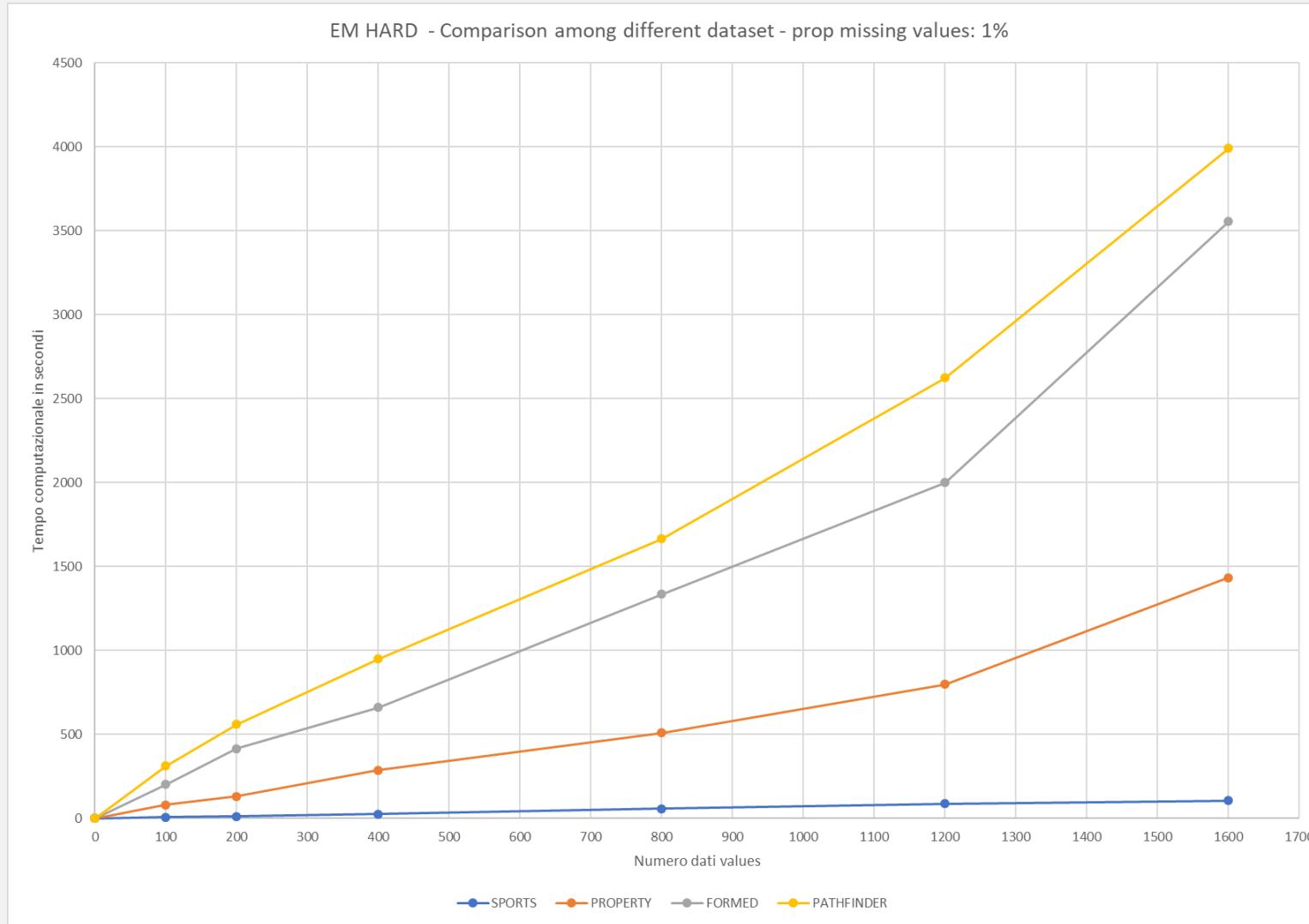
## Analisi:

- In **EM HARD** il passo di **Expectation** (inferenza esatta) rappresenta in media il 96% del tempo di computazionale totale, **Preprocessing** 1,7% e **Maximisation (inc. expected sufficient statistics)** 2,3%.
- In **EM SOFT** il passo di **Expectation** (inferenza esatta) rappresenta in media il 92,5% del tempo di computazionale totale, **Preprocessing** 1,2% e **Maximisation (inc. expected sufficient statistics)** 6,3%.

# TEMPO COMPUTAZIONALE - SPORTS



# TEMPO COMPUTAZIONALE - CONFRONTO



# TEMPO COMPUTAZIONALE - SPORTS

A causa di una dimenticanza, nel settaggio della working directory in RStudio, non è stato possibile calcolare e computare i tempi computazionali per EM SOFT.

I risultati saranno disponibili a breve.