

PERFORMANCE EVALUATION PROCESS ALGEBRA



Progetto di: Polverelli Enrico

Linguaggi e Modelli Computazionali LM

a.a. 2011/2012

SOMMARIO

- Introduzione a PEPA
- Sintassi e semantica degli operatori
- Il plugin Eclipse per PEPA: un caso di studio
- Fluid Flow Approximation:
 - Motivazioni
 - Generazione del sistema di ODE
 - Caso di studio relativo ad un attacco DDOS
- Conclusioni



INTRODUZIONE A PEPA

- Tecnica di descrizione basata sulla classica algebra di processo, migliorata con informazioni temporali stocastiche.
- Gli elementi base sono i *componenti* e le *attività*.
- Ogni attività è rappresentata da una coppia (α, r) :
 - α è il tipo di azione (o τ per azione non definita)
 - r è il rate dell'azione, cioè il parametro della distribuzione esponenziale negativa che ne regola la durata (o T per rate passivo)
- I *componenti* e le *attività* corrispondono agli *stati* e alle *transizione* della sottostante CTMC, mapping possibile grazie alla distribuzione esponenziale negativa del parametro r .
- Ogni attività (α, r) ha una probabilità di essere completata entro un tempo t pari ad $1 - e^{-rt}$



SINTASSI E SEMANTICA

- La sintassi del linguaggio è la seguente:

$$\begin{aligned} S &::= (\alpha, r).S \mid S + S \mid C_S \\ P &::= P \boxtimes_L P \mid P/L \mid C \end{aligned}$$

- La semantica degli operatori del linguaggio:

Prefix

$$\frac{}{(\alpha, r).E \xrightarrow{(\alpha, r)} E}$$

Choice

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E + F \xrightarrow{(\alpha, r)} E'} \quad \frac{F \xrightarrow{(\alpha, r)} F'}{E + F \xrightarrow{(\alpha, r)} F'}$$

Cooperation ($\alpha \notin L$)

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E \boxtimes_L F \xrightarrow{(\alpha, r)} E' \boxtimes_L F}$$

Cooperation ($\alpha \in L$)

$$\frac{E \xrightarrow{(\alpha, r_1)} E' \quad F \xrightarrow{(\alpha, r_2)} F'}{E \boxtimes_L F \xrightarrow{(\alpha, R)} E' \boxtimes_L F'}$$

Hiding

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E/L \xrightarrow{(\alpha, r)} E'/L} \quad (\alpha \notin L)$$

$$\frac{E \xrightarrow{(\alpha, r)} E'}{E/L \xrightarrow{(\tau, r)} E'/L} \quad (\alpha \in L)$$

Constant

$$\frac{E \xrightarrow{(\alpha, r)} E'}{C \xrightarrow{(\alpha, r)} E'} \quad (C \stackrel{\text{def}}{=} E)$$

$$\frac{F \xrightarrow{(\alpha, r)} F'}{E \boxtimes_L F \xrightarrow{(\alpha, r)} E \boxtimes_L F'}$$

$$R = \frac{r_1}{r_\alpha(E)} \frac{r_2}{r_\alpha(F)} r_m$$

$$r_m = \min(r_\alpha(E), r_\alpha(F))$$

PEPA ECLIPSE PLUGIN

- Per mostrare quello che è possibile fare con il PEPA Eclipse plugin è stato usato il seguente modello di un web service bancario:

```
Customer_Idle = (request , r_request ).Customer_Entering;  
Customer_Entering = (enterData,r_enterdata).Customer_Upload;  
Customer_Upload = (uploadData, r_upload ).Customer_Waiting;  
Customer_Waiting = (approve, T).Customer_Idle + (decline,T).Customer_Deciding;  
Customer_Deciding = (reapply, r_reapply).Customer_Entering + (reapply,  
r_reapply).Customer_Idle;  
  
Service_Idle = (uploadData, T ).Service_Validating;  
Service_Validating = (validateData, r_validate ).Service_Sending;  
Service_Sending = (sendBank, r_sendBank ).Service_Idle;  
  
Bank_Idle = (sendBank, T ).Bank_PreDecide;  
Bank_PreDecide = (predecide, r_predecide).Bank_Approve + (predecide,  
r_predecide).Bank_Decline + (predecide, r_predecide).Bank_Employee;  
Bank_Employee = (decide, r_decide).Bank_Confirm + (decide, r_decide).Bank_Decline;  
Bank_Confirm = (decide, r_decide).Bank_Approve + (decide, r_decide).Bank_Decline;  
Bank_Approve = (approve, r_inform).Bank_Idle;  
Bank_Decline = (decline, r_inform).Bank_Idle;  
  
Customer_Idle <uploadData, approve, decline> (Service_Idle <sendBank> Bank_Idle)
```



FUNZIONALITÀ DEL PLUGIN ECLIPSE

Grazie all'uso del plugin, sul modello è stato possibile effettuare le seguenti operazioni:

- Derivazione dello spazio degli stati
- Steady state analysis
- Misure di performance:
 - Throughput
 - Population
 - Utilisation
- Analisi temporale del sistema
- Esperimenti con i rate delle azioni
- Model Checking



FLUID FLOW APPROXIMATION

- Necessità di modelli sempre più accurati per avere misure di performance più accurate.
- Modelli complessi implicano l'esplosione dello spazio degli stati del modello.
- Necessità di una tecnica che permetta di poter analizzare modelli di sistemi complessi e massicciamente paralleli.
- L'approccio ad ODE permette di ridurre un sistema complesso in un sistema di equazioni differenziali molto più facile da risolvere.
- Possibile solo per sistemi altamente simmetrici.



GENERAZIONE DELLE ODE

- Per prima cosa lo spazio degli stati deve essere ridotto tramite aggregazione.
- A questo punto l'idea è che la variazione dinamica di ogni derivative del modello sia esprimibile come:

$$\begin{aligned} \frac{dv_{ij}(t)}{dt} = & - \underbrace{\sum_{k: C_{ij} \xrightarrow{(a, \cdot)} C_{ik}} p_a(C_{ij}, C_{ik}) \rho_a(C_{ij}, P(t)) v_a(C_{ij}, P(t))}_{\text{exit activities}} \\ & + \underbrace{\sum_{k: C_{ik} \xrightarrow{(b, \cdot)} C_{ij}} p_b(C_{ik}, C_{ij}) \rho_b(C_{ik}, P(t)) v_b(C_{ik}, P(t))}_{\text{entry activities}} \end{aligned}$$


GENERAZIONE DELLE ODE

Ogni sommatoria è composta dal prodotto di tre termini:

1. La probabilità che uno stato evolva in un altro al seguito dell'esecuzione dell'azione
2. Rate osservato localmente ad un certo componente all'interno del sistema
3. Numero di componenti che quell'azione fa evolvere

Ogni sommatoria sarà con il segno positivo se riferita ad un'attività di entrata, negativo se riferita ad un'attività di uscita



CASO DI STUDIO

- Modello che descriva l'evoluzione di un sistema *Internet-scale* sottoposto ad un attacco DDOS.

```
S = (infectS, 10.0).I;
```

```
I = (infectI, beta).I + (infectS,10.0).I + (patch, gamma).R + (attack,  
x).A;
```

```
A = (attackA, lambda).A + (patch, gamma).R;
```

```
R = (stop,1.0).R;
```

```
Net = (infectI,10.0).Net0 + (attackA, 1.0).Net1;
```

```
Net0 = (infectS, beta).Net + (failR, delta).Net;
```

```
Net1 = (attackV, p).Net + (failR, delta).Net;
```

```
V = (attackV, 1.0).V0;
```

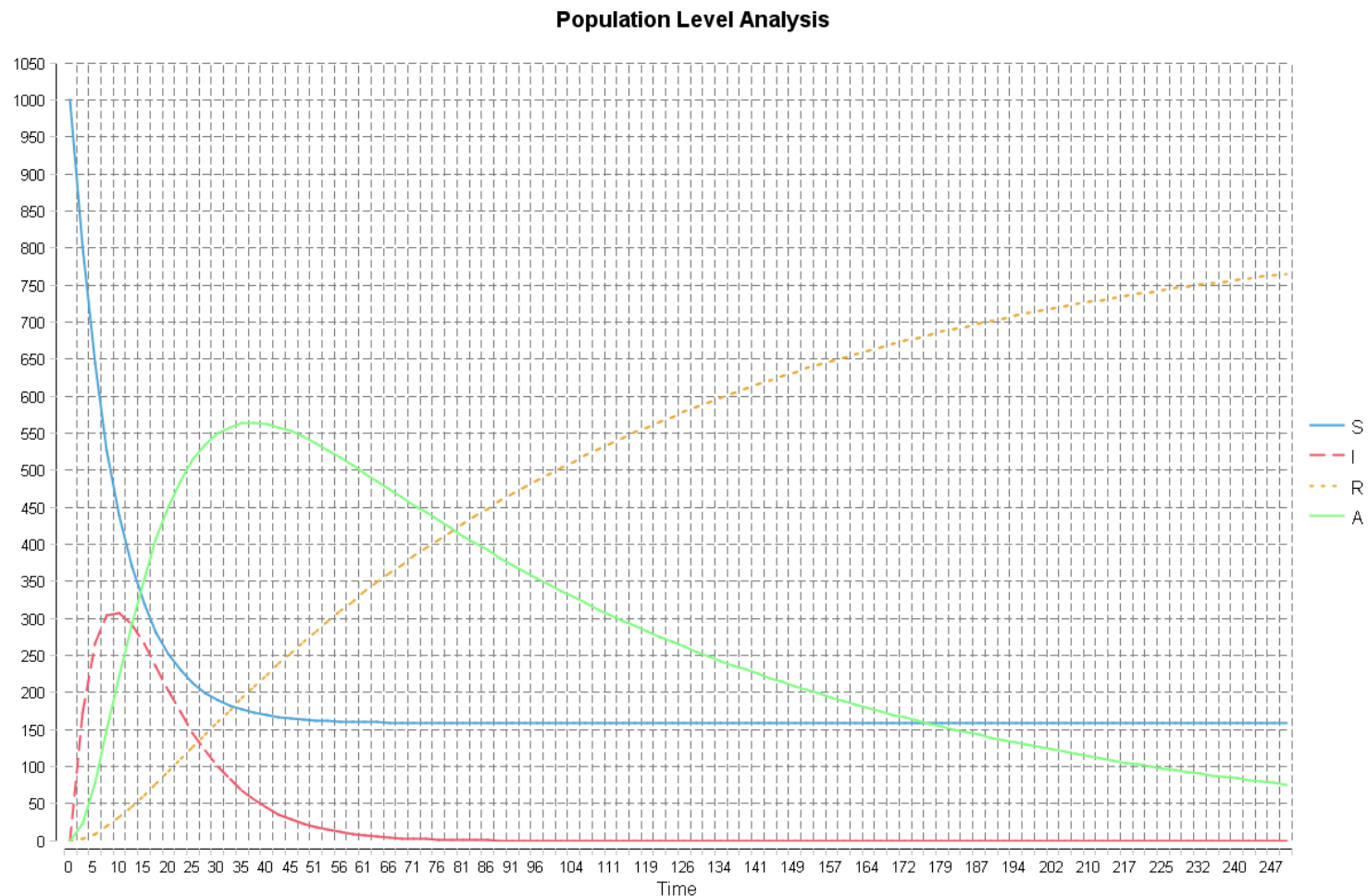
```
V0 = (release, sigma).V;
```

```
(S[1000] <> I[10] <> V) <infectI, infectS, attackA, attackV> Net[700]
```



CASO DI STUDIO

- L'analisi ad ODE produce il seguente risultato



PEPA - CONCLUSIONI

○ PRO:

- Tecnica di descrizione utile per descrivere sistemi complessi
- Basso numero di operatori, sintassi semplice
- Alta espressività grazie alla possibilità di esprimere informazioni temporali sulle azioni

○ CONTRO:

- Difficoltà nella parametrizzazione dei modelli



PEPA ECLIPSE PLUGIN - CONCLUSIONI

○ PRO:

- Tool unico per effettuare diverse analisi di performance.
- Linguaggio che ricalca fedelmente il linguaggio PEPA.
- Possibilità di esportare risultati in altri tool.

○ CONTRO:

- Scale non significative nei grafici.
- Non supporta i rate passivi durante le analisi ODE.
- Poco intuitivo in certe fasi.



FLUID FLOW APPROXIMATION - CONCLUSIONI

○ PRO:

- Approccio fondamentale per trattare sistemi complessi.
- Possibilità di prevedere il comportamento del sistema già in fase di progetto.
- Molto performante.

○ CONTRO:

- Possibile solo su certi tipi di sistemi altamente simmetrici, risultati non accurati negli altri casi



SVILUPPI FUTURI

- *Stochastic Differential Equation*
- Approccio ibrido discreto/continuo

