MODELLI AD AUTOMI CELLULARI PER LA SIMULAZIONE DI FENOMENI MACROSCOPICI COMPLESSI

Salvatore Di Gregorio

Dept. of Mathematics & Center of Excellence for High Performance Computing
University of Calabria, Arcavacata, I-87036 Rende (CS), Italy

EMPEDOCLES RESEARCH GROUP
toti.dig@unical.it

Sommario:

- **Automi Cellulari per modellare fenomeni macroscopici complessi di tipo acentrato: un approccio pragmatico**
- **Algoritmo di minimizzazione delle differenze**
- **Esempi di applicazioni:**
  - **colate di detrito : SCIDDICA S3hex**
  - **traffico autostradale : STRATUNA**
- **Commenti**

CRITERI DEFINITORI DI *AC* PER FENOMENI MACROSCOPICI

Si propone qui una definizione estesa di *Automi Cellulari* (*AC*) per modellare fenomeni macroscopici di tipo acentrato.
Formalmente si definisce un *AC* come la settupla:

$$\langle R, G, S, X, P, \tau, \gamma \rangle$$

PARAMETRI GLOBALI

Primo requisito: L'*AC* astratto deve essere correlato in modo univoco al fenomeno macroscopico reale riguardo tempo e spazio.

- Alcuni **parametri globali** debbono essere considerati:
  almeno le dimensioni della cella ovvero la distanza fra i centri di due celle vicine p_d
  e il tempo corrispondente ad un passo della funzione di transizione p_t;

- $P = (p_d, p_t, \ldots.)$ è l'insieme finito dei parametri globali, che influiscono sulla funzione di transizione.

CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (1)

- La cella corrisponde normalmente ad una porzione di spazio; quindi lo spazio cellulare dovrebbe essere tridimensionale:

$$R = \{(x, y, z) \mid x, y, z \in N,\ 0 \le x \le l_x\text{-}1,\ 0 \le y \le l_y\text{-}1,\ 0 \le z \le l_z\text{-}1\}$$

. è l'insieme di punti a coordinate intere, che definisce la regione finita dello spazio, dove il fenomeno evolve. *N* è l'insieme dei numeri naturali.

Se sono lecite semplificazioni, è banale ridurre la formula a 1-2 dimensioni.

- In una regione finita le celle ai bordi non hanno vicinato completo. Per ovviare a ciò si consideri una variazione della definizione della funzione *V*:

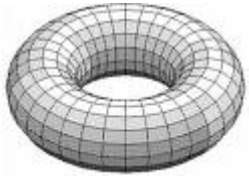
$X=\{\xi_0, \xi_1, \ldots\ldots \xi_{m-1}\}$, la relazione (o indice) di vicinanza, è un insieme finito di vettori tridimensionali, che definisce l'insieme $V(X,i,l)$ di vicini della cella generica $i=<i_x, i_y, i_z>$, con $l=<l_x, l_y, l_z>$, come segue tramite la funzione *W*

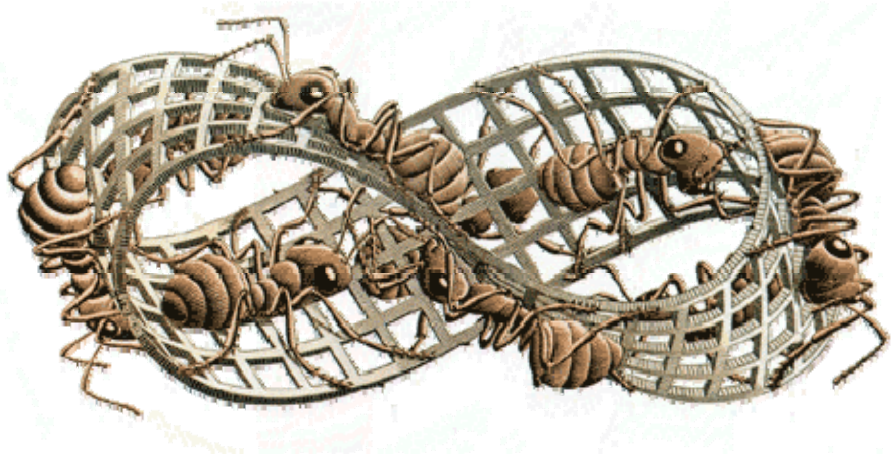
sia $m=\#X$, allora $V(X,i,l) = \{W(i,\xi_0,l), W(i,\xi_1,l), \ldots W(i,\xi_{m-1},l)\}$

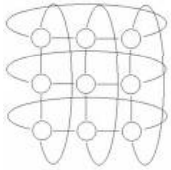
con ξ_0 il vettore nullo

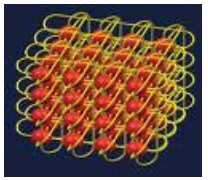
e $V(i,\xi_j,l) = < (i_x+ \xi_{jx}) mod\ l_x, (i_y+ \xi_{jy}) mod\ l_y, (i_z+ \xi_{jz}) mod\ l_z>$ $\quad 0 \le j \le m\text{-}1$

- Visualizzazioni









CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (2)

SOTTOSTATI

Secondo requisito: **la macroscopicità del fenomeno può implicare eterogeneità.**

Ciascuna caratteristica, rilevante all'evoluzione del sistema e relativa alla porzione di spazio corrispondente alla cella, è individuato come un sottostato; l'insieme finito ***S*** degli stati è espresso dal prodotto Cartesiano degli insiemi dei sottostati:

$$S=S_1\times S_2\times.....\times S_n\ ;$$

• Il valore di un sottostato è costante nello spazio occupato dalla cella (ad esempio la temperatura);

Quando una caratteristica (ad es. una quantità fisica) è espressa come una variabile continua, un finito, ma sufficiente numero di cifre significative sono usate, così che l'insieme dei valori possibili può essere arbitrariamente grande ma finito.

Lo spazio cellulare dovrebbe essere tridimensionale, ma una riduzione a due dimensioni è permesso se le quantità attinenti alla terza dimensione (l'altezza) possono essere rappresentate come sottostatì della cella.

CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (3)

PROCESSI "ELEMENTARI"

Terzo requisito: Così come lo stato della cella può essere decomposto in sottostati, la funzione di transizione τ può essere suddivisa in processi "elementari", definiti dalle funzioni $\sigma_1, \sigma_2 .... \sigma_p$ con p il numero di processi elementari.

- I processi elementari sono applicati sequenzialmente secondo un ordine definito. Differenti processi elementari possono comportare un vicinato differente. Ciascun processo elementare aggiorna gli stati dell'AC.

- La variazione di un sottostato è dato da:

  trasformazioni interne
  interazioni nel vicinato

- Le trasformazioni interne sono definite come variazioni nei valori dei sottostati determinate da condizioni interne alla cella (solo sottostati della cella).

- Le interazioni locali sono definite come variazioni determinate da interazioni dei sottostati nel vicinato.

CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (4)

VICINATO

- Il vicinato dell'*AC* è specificato dall'unione di tutti i vicinati associati a ciascun processo elementare,
- Siano specificati ***p*** processi elementari, allora per il j-simo processo, $\mathbf{1 \le j \le p}$, si ha :

$$\sigma_j : Q_j^{m_j} \rightarrow Q_j^{'}$$

dove $\boldsymbol{Q_j}$ and $\boldsymbol{Q_j}^{'}$ sono i prodotti Cartesiani degli elementi dei sottoinsiemi di $\boldsymbol{Q}$, con

$$\boldsymbol{Q}=\{\boldsymbol{S_1},\boldsymbol{S_2},.....,\boldsymbol{S_n}\},$$

con $\boldsymbol{m_j}$ il numero di celle del vicinato $\boldsymbol{X_j}$, attinente al processo elementare *j*;

$\boldsymbol{Q_j}$ individua i sottostati nel vicinato, che influiscono sulla variazione di valore dei sottostati e $\boldsymbol{Q_j}^{'}$ individua i sottostati della cella che cambiano di valore.

- ***X***, la relazione di vicinato dell'AC è definita come segue:

$\boldsymbol{X} = \{\boldsymbol{X_1} \cup \boldsymbol{X_2} \cup ... \cup \boldsymbol{X_p}\}$, dove $\boldsymbol{X_j}$ $\mathbf{1 \le j \le p}$ è riferito al processo elementare *j*-simo.

CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (5)

INFLUENZE ESTERNE

Quarto requisito: Talvolta, una sorta di input dal "mondo esterno" sulle celle dell'*AC* deve essere considerato; esso tiene conto di influenze esterne che non possono essere descritte in termini acentrati (per esempio l'effusione di lava ai crateri) oppure per un qualche approccio probabilistico al fenomeno. Quindi funzioni speciali e/o addizionali (γ) debbono essere specificate per quel tipo di celle (G). NB: γ e G non sempre sono necessarie da specificare nell'AC.

- $G = \{G_1 \cup G_2 \cup ... \cup G_n\}$ è l'insieme di celle, che subiscono influenze dal "mondo esterno"; qui sono considerate n influenze esterne, ciascuna definisce una sottoregione G_i $1 \leq i \leq n$ dello spazio cellulare, con $G \subseteq R$.

- $\gamma: N \times G \rightarrow S$ esprime le influenze esterne per le celle di G nello spazio cellulare; essa determina la variazione dello stato S per le celle in G.
  N, l'insieme dei numeri naturali, è qui riferito ai passi dell'*AC*.
  γ è specificato dall' applicazione sequenziale di n funzioni
  $\gamma_1: N \times G_1 \rightarrow Q_1$, $\gamma_2: N \times G_2 \rightarrow Q_2$, … $\gamma_v: N \times G_n \rightarrow Q_n$ dove Q_1 , Q_2 ... Q_n sono prodotti Cartesiani di elementi dei sottoinsiemi di Q, $Q=\{S_1, S_2, ..., S_n\}$.

CRITERI DEFINITORI DI *AC*: $<R, G, S, X, P, \tau, \gamma>$ (6)

ESTENSIONE DELLA NOZIONE DI STATO QUIESCENTE

Uno stato quiescente q è tale che $\tau(q, q.....q) = q$

Una configurazione c è finita se $c(i)=q$ eccetto un numero finito di celle.

- In una configurazione finita c, possono cambiare stato soltanto le celle che non appartengono alla regione stazionaria R_s definita come:

$$R_s = \{i \mid (c(i) = q) \wedge (j \in c(N(X,i)) \Rightarrow c(j) = q)\}$$

Si introduce una estensione della nozione di stato quiescente, sia $c' = \tau(c)$:

- un insieme di stati $S_0 \subset S$ si definisce inattivo se $N(X,i) \in S_0^m \Rightarrow c(i)=c'(i)$.
- Uno stato s si definisce inerte se $c(i)=s \Rightarrow c(i)=c'(i)$;

  S_{inerte} è l'insieme degli stati inerti.
- la regione stazionaria R_s di una configurazione c è definita:

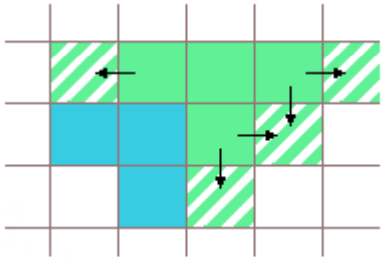
$$R_s = \{i \mid (c(i) \in S_{inerte}) \vee (j \in c(N(X,i)) \Rightarrow j \in S_0)\}.$$

- Celle con stati inattivi non cambiano stato fino a quando il vicinato rimane anch'esso inattivo, mentre le celle con stati inerti non cambiano mai di stato; quindi è importante sviluppare metodi per gestire la regione stazionaria per ottimizzare i tempi di calcolo, anche se l'effettivo sincronismo di calcolo può essere indebolito in tali condizioni.

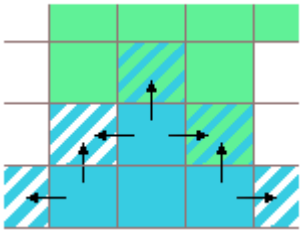
WaTor scritto in termini di AC macroscopici: le regole

- **WA-TOR** di A.K. Dewdney è un AC toroidale a celle quadrate, che rappresenta un mondo acquatico, dove squali e pesciolini interagiscono.

- Un pesciolino nuota a caso verso una cella libera adiacente. Se per un certo numero di passi sopravvive, allora si riprodurrà nella prima cella libera adiacente .

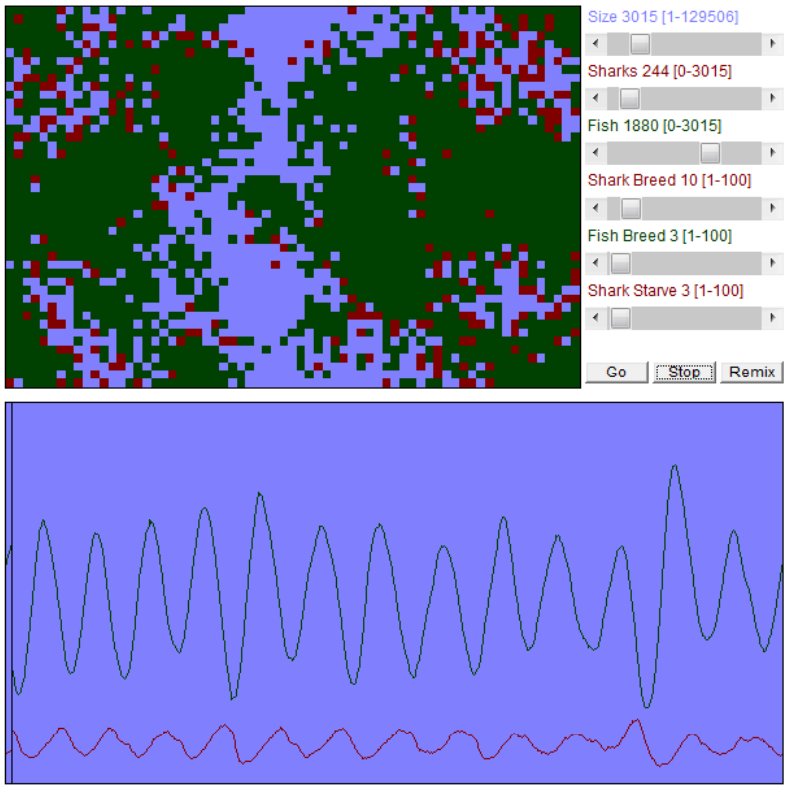


- Uno squalo mangia un pesciolino, se si trova in una cella adiacente, altrimenti nuota verso una cella libera. Se uno squalo non mangia per un certo numero di passi, muore. Uno squalo si riproduce con le stesse modalità di un pesciolino.



WaTor scritto in termini di AC macroscopici: il comportamento

- Se gli squali sono pochissimi, i pesciolini si moltiplicheranno velocemente, seguirà una rapida crescita del numero degli squali per la facilità di procurarsi cibo.
- Quando i pesciolini si ridurranno per il gran numero degli squali, anche gli squali diminuiranno dopo un po' per mancanza di cibo.



- N.B. questa è una specificazione incompleta di Wa-Tor, le cui regole possono essere interpretate ed approssimate in vari modi.

  Nel seguito diamo una formalizzazione di WaTor, più precisa da quella del sito sottocitato, la quale permette la conservazione del numero di pesci negli spostamenti:

http://www.leinweb.com/snackbar/wator/

WaTor scritto in termini di AC macroscopici: una formalizzazione

L'AC di WaTor è bidimensionale a celle quadrate con vicinato di von Neumann:

$$\text{WaTor} \equiv < R_{toroidale} = \{(x, y) \mid x, y \in N, 0 \le x \le l_x\text{-}1, 0 \le y \le l_y\text{-}1\},$$
$$X = \{x,y|(|x|+|y| \le 1)\},$$
$$P = \{p_{inedia}, p_{matur\_s}, p_{matur\_p}\},$$
$$S = S_{TIPO} \times S_{ETÀ} \times S_{DIR} \times S_{DIGIUNO},$$
$$\tau \equiv \tau_{nutrizione}, \tau_{riproduzione}, \tau_{spostamento}$$
$$>$$

	1 (0,1)	
3 (-1,0)	**0** (0,0)	**2** (1,0)
	4 (0,-1)	

Saranno dati in seguito segmenti di codice in pseudo-C++

NB: la matrice relativa al sottostato S_Q è specificata come *mat_Q*

NB: nei segmenti di programma relativi alla funzione di transizione

- **il valore del sottostato S_Q è espresso con Q o con $Q[i]$, $0 \le i \le 4$, quando la specificazione del sottostato nelle celle del vicinato è necessario.**
- **ΔQ significa variazione del valore del sottostato S_Q.**
- ***nQ* significa nuovo valore del sottostato S_Q.**
- **il valore del parametro p_q è espresso con q.**

WaTor scritto in termini di AC macroscopici: una formalizzazione

L'AC di WaTor è bidimensionale a celle quadrate con vicinato di von Neumann:

$\mathbf{WaTor} \equiv <R_{toroidale} = \{(x, y) \mid x, y \in N, 0 \le x \le l_x\text{-}1, 0 \le y \le l_y\text{-}1\}, X = \{x,y|(|x|+|y| \le 1)\},$
$P = \{p_{inedia}, p_{matur\_s}, p_{matur\_p}\}, S = S_{TIPO} \times S_{ETÀ} \times S_{DIR} \times S_{DIGIUNO}, \tau \equiv \tau_{nutrizione}, \tau_{riproduzione}, \tau_{spostamento} >$

TIPO può avere valore **0** o '***m***'(mare), **1** o '***p***'(pesciolino) e **2** o '***s***'(squalo).

DIR -ezione prende i valori degli indici di vicinato, **0** indica direzione nulla, in più il valore -1 viene utilizzato per inibire lo spostamento.

inedia è la costante "numero intero di passi" (associata al sottostato ***DIGIUNO***), senza che lo squalo mangi, bastante a determinare la morte dello squalo per inedia.

matur_s, matur_p sono le costanti "numero intero di passi" (associate al sottostato ***ETÀ***), necessario agli squali ed i pesciolini per raggiungere la maturità nella riproduzione.

- $\tau \equiv \tau_{nutrizione}, \tau_{riproduzione}, \tau_{spostamento}$ sono i tre processi elementari di τ, a loro volta suddivisi in sottoprocessi elementari:
  - $\tau_{nutrizione} \equiv \tau_{ricerca\_cibo}, \tau_{risol\_conflitti\_cibo}, \tau_{cattura\_cibo}, \tau_{decesso\_per\_inedia}$
  - $\tau_{riproduzione} \equiv \tau_{ricerca\_cella\_nido}, \tau_{risol\_conflitti\_ripr}, \tau_{nascita}$
  - $\tau_{spostamento} \equiv \tau_{ricerca\_direz}, \tau_{risol\_conflitti\_spost}, \tau_{movimento}$

	1 (0,1)	
3 (-1,0)	**0** (0,0)	**2** (1,0)
	4 (0,-1)	

una formalizzazione di WaTor:

INIZIALIZZAZIONE

È necessario specificare inizialmente la percentuale di celle di **TIPO** mare, pesciolino, squalo e per quelle di **TIPO** pesciolino e squalo bisogna inizializzare i valori **ETÀ** , per quelle di **TIPO** squalo anche i valori di **DIGIUNO**, i valori indefiniti vengono inizializzati a **0**.

A lato un segmento di programma in pseudo-C++, ove abbiamo l'inizializzazione delle matrici dei sottostati con circa 60% di celle di ***TIPO*** mare, 5% di ***TIPO*** squalo ed il restante 35% di ***TIPO*** pesciolino.

Bisogna pure specificare i valori dei parametri, che sono costanti durante l'esecuzione del programma; ad esempio:

```
… …
const int inedia = 15;
const int matur_s = 20;
const int matur_p = 5;
… …
```

```
… …
for (x=0 ; i<l_x ; x++ )
    for (y=0 ; i<l_y ; y++ )
     {  srand(time (0));
        a_caso_1_100 = 1+rand()%100;
        if  ( a_caso_1_100 <= 60 )
                { mat_ TIPO [x][y] = 0;
                  mat_ DIR [x][y]= 0;
                  mat_ ETÀ [x][y] = 0;
                  mat_ DIGIUNO [x][y]= 0;
                }
        else
        if  ( a_caso_1_100 > 95 )
                { mat_ TIPO [x][y] = 2;
                  mat_ DIR [x][y]= 0;
                  mat_ ETÀ [x][y] = 0;
                  mat_ DIGIUNO [x][y]= 0;
                }
        else
                { mat_ TIPO [x][y] = 1;
                  mat_ DIR [x][y]= 0;
                  mat_ ETÀ [x][y] = 0;
                  mat_ DIGIUNO [x][y]= 0;
                }
      }
… …
```

una formalizzazione di WaTor: funzioni ausiliarie notevoli

"scegli_a_caso"

- i sottoprocessi elementari del tipo "**ricerca_...**" sono specificati tramite la funzione ausiliare **scegli_a_caso**. Nel caso di uno squalo che cerchi una cella occupata da un pesciolino per nutrirsi oppure un pesce che cerchi una cella libera dove riprodursi o dove spostarsi, questa funzione trova (se esiste) la prima cella vicina "idonea" a partire da un valore casuale di indice di vicinato fra **1** e **4**.
- L'indice di questa cella specifica il valore del sottostato ***DIR***, cioè la direzione idonea per una specifica azione, altrimenti ritorna **0**, l'indice della cella centrale.

```
int   scegli_a_caso  (int sottostato[] , int valore)
{     int  vicino, cont=0;
      srand(time (0));
      vicino=1+rand()%4;
      while ( ! ((sottostato[vicino]==valore)) || (cont==4)))
        {       cont++;
                vicino=vicino%4+1;
        }
      if  (sottostato[vicino]==valore)   return (vicino) ;
      else   return 0;
}
```

una formalizzazione di WaTor: funzioni ausiliarie notevoli

"risolvi_conflitto"

- i sottoprocessi elementari del tipo "**risol_conflitti_...**" sono specificati tramite la funzione ausiliare **"risolvi_conflitto",** scritto sulla falsariga di **scegli_a_caso.** Nel caso si verifichino situazioni di conflitto: più di uno squalo a voler mangiare lo stesso pesciolino, più pesci che si vogliono muovere o riprodurre nella stessa cella, vale a dire più vicini ***i*** con ***DIR***[***i***]=5-***i*** , allora si sceglie a caso il primo vicino dando a ***DIR*****[0]** il valore dell'indice del vicino prescelto, ciò ha il significato di una selezione.

```
int  risolvi_conflitto ( )
 {   srand(time (0));
     vicino=1+rand()%4;
      while ( ! ((DIR[vicino]==5-vicino)) || (cont==4)))
       {      cont++;
              vicino=vicino%4+1;
        }
     if  (DIR[vicino]==(5-vicino))   return (vicino) ;
     else return 0;
   } … …
```

	1 DIR=4	
3 DIR=2	**0** DIR=?	**2** DIR=2
	4 DIR=1	

una formalizzazione di WaTor: $\tau_{nutrizione}$

ricerca_cibo verifica per la cella centrale (indice **0**) di ***TIPO*** squalo/**2**, se un'altra cella del suo vicinato è di ***TIPO*** pesciolino/**1**.

```
/* τricerca_cibo */
… …
nDIR[0]=0;
if  (TIPO[0]==2)
 {  indice = scegli_a_caso  (TIPO[] , 1);
   nDIR[0]=indice;
 }
… …
```

risol_conflitto_cibo verifica per la cella centrale (indice **0**) di ***TIPO*** pesciolino/**1** se un'altra cella del suo vicinato abbia un valore di ***DIR*** $\neq 0$ (significa che è di ***TIPO*** squalo/**2**) e, se, vi è più di uno squalo che punta alla cella col pesciolino, a quale deve essere permesso di mangiare il pesciolino.

```
/* τrisol_conflitto_cibo */
… …
nDIR[0]= 0;
if  (TIPO[0]==1)
  {  scelto= risolvi_conflitto();
     nDIR[0]=scelto ;
  }
… …
```

una formalizzazione di WaTor: $\tau_{nutrizione}$

cattura_cibo sostituisce nelle celle con ***TIPO*=1** (pesciolino) con ***DIR*≠0**, con ***nTIPO*=2** (squalo); sostituisce nelle celle con ***TIPO*=2** (squalo) con ***DIR*≠0**, ***nTIPO*=0** (mare) .

```
/* τcattura_cibo */
… …
nTIPO[0]=TIPO[0];
nDIGIUNO[0]=DIGIUNO[0];
nDIR[0]=0;
direz= DIR[0];
if  ((TIPO[0]==2) && (DIR[0]<>0)&&(DIR[direz]=5-direz))
        nTIPO[0]=0;
if  ((TIPO[0]==1) && ((DIR[0]<>0 )&&(DIR[direz]=5-direz))
{  nTIPO[0]=2);
   nDIGIUNO[0]=0;
   nDIR[0]=-1;
} … …
```

decesso_per_inedia verifica per la cella centrale (indice **0**) di ***TIPO*** squalo (**2**) se il valore di ***DIGIUNO* > *inedia*** la cella diventi di ***TIPO*** acqua.

```
/* τdecesso_per_inedia */
… …
if  ((TIPO[0]==2) && (DIGIUNO[0]>inedia))     nTIPO[0]=0 ;
else  nTIPO[0]=TIPO[0];
… …
```

una formalizzazione di WaTor: $\tau_{riproduzione}$

- **ricerca_cella_nido** verifica se la cella centrale (indice **0**) è di ***TIPO*** squalo/**2** o di ***TIPO*** pesciolino/**1**, se è stata raggiunta la maturità di riproduzione e se questa è possibile in un'altra cella del vicinato di ***TIPO*** mare/**0**.

```
/* τ ricerca_cella_nido */
… …
nDIR[0]=0;
if  (((TIPO[0]==2) && (ETÀ[0]>=matur_s)) || ((TIPO[0]==1) && (ETÀ[0]>=matur_p)))
{  indice = scegli_a_caso  (TIPO[] , 0);
   nDIR[0]=indice;
}  … …
```

- **risol_conflitti_ripr** verifica per la cella centrale (indice **0**) di ***TIPO*** mare/**0** se un'altra cella del suo vicinato abbia un valore di ***DIR*** $\neq 0$ (significa che è di ***TIPO*** pesciolino/**1** o squalo/**2**) e sceglie, se vi è più di uno squalo e/o pesciolino che punta alla cella, a quale deve essere permesso di riprodursi.

```
/* τ risol_conflitti_ripr */
… …
nDIR[0]= 0;
if  (TIPO[0]==0)
    { scelto= risolvi_conflitto();
       nDIR[0]=scelto ;
    } … …
```

una formalizzazione di WaTor: $\tau_{riproduzione}$

- **nascita** permette la nascita nelle celle con ***TIPO* 0**/mare con ***DIR*≠0**, cambiando il ***TIPO* 0**/mare in ***TIPO* 1**/pesciolino o ***TIPO* 2**/squalo; azzera l'età dello squalo-pesciolino riprodottosi e quello generato, inibisce i loro movimenti ed azzera il loro sottostato ***DIGIUNO***.

```
/* τnascita */
… …
nTIPO[0]=TIPO[0];
nETÀ [0]=ETÀ [0];
nDIR[0]=0;
direz= DIR[0];
nDIGIUNO[0]=DIGIUNO[0];
if  ((TIPO[0]==0) && (DIR[0]<>0)&&(DIR[direz]=5-direz))
{  nTIPO[0]=TIPO[5-direz]);
   nDIGIUNO[0]= 0,
   nETÀ [0]= 0;
   nDIR[0]=-1;
}
if  (((TIPO[0]==1) || (TIPO[0]==2)) && ((DIR[0]<>0 )&&(DIR[direz]=5-direz))
{  nETÀ [0]= 0;
   nDIGIUNO[0]=0;
   nDIR[0] =-1;
} … …
```

una formalizzazione di WaTor: $\tau_{spostamento}$

- **ricerca_cella_libera** verifica se la cella centrale (indice **0**) è di ***TIPO*** squalo/**2** o di ***TIPO*** pesciolino/**1** il cui movimento non è inibito (***DIR*≠-1**), se esiste nel vicinato una cella di ***TIPO*** mare/**0** dove muoversi.

```
/* τ ricerca_cella_libera */
... ...
nDIR[0]=DIR[0];
if  (((TIPO[0]==2) && (DIR[0]<>-1)) || ((TIPO[0]==1) && (DIR[0]<>-1)))
{  indice = scegli_a_caso  (TIPO[] , 0);
   nDIR[0]=indice;
}  ... ...
```

- **risol_conflitti_spost** verifica per la cella centrale (indice **0**) di ***TIPO*** mare/**0** se un'altra cella del suo vicinato abbia un valore di ***DIR*** > 0 (significa che è di ***TIPO*** pesciolino/**1** o squalo/**2** non inibito) e sceglie, se vi è più di uno squalo e/o pesciolino che punta alla cella, a quale deve essere permesso di spostarsi.

```
/* τ risol_conflitti_ripr */
... ...
nDIR[0]= 0;
if  (TIPO[0]==0)
     {  scelto= risolvi_conflitto();
        nDIR[0]=scelto ;
     } ... ...
```

una formalizzazione di WaTor: $\tau_{spostamento}$

- **movimento** permette lo "spostamento" dello squalo-pesciolino verso le celle di ***TIPO* 0**/mare con ***DIR*≠0**, cambiando il ***TIPO* 0**/mare in ***TIPO* 1**/pesciolino o ***TIPO* 2**/squalo, mentre il corrispondente ***TIPO* 1**/pesciolino o ***TIPO* 2**/squalo diventa ***TIPO* 0**/mare. n***DIR*=0**;per tutte le celle di WaTor.

```
/* τmovimento */
… …
nTIPO[0]=TIPO[0];
nETÀ [0]=ETÀ [0];
nDIR[0]=0;
direz= DIR[0];
nTIPO[0]=TIPO[0];
if  ((TIPO[0]==0) && (DIR[0]<>0)&&(DIR[direz]=5-direz))
      nTIPO[0]=TIPO[5-direz]);
if  (((TIPO[0]==1) || (TIPO[0]==2)) && ((DIR[0]<>0 )&&(DIR[direz]=5-direz))
      nTIPO[0]=0;
… …
```

Algoritmo di minimizzazione delle differenze (1)

- ***Molti sistemi complessi evolvono a livello locale verso condizioni di massimo equilibrio possibile: in termini di AC il sistema tende a minimizzare nel vicinato le differenze relative ad una certa grandezza, dando luogo a flussi di certe quantità fra le celle.***

- **Problema:** Determinare i flussi di una quantità q dalla cella centrale alle altre n celle del vicinato che ne minimizzi le differenze:

- **Definizioni:** q_d = quantità nella cella centrale, che può essere distribuita,
  q_0 = quantità inamovibile nella cella centrale
  q_i = quantità nella cella i $\quad 1 \le i \le n$
  f_i = flussi dalla cella centrale alla cella i $\quad 1 \le i \le n$
  f_0 è la parte di q_d che resta nella cella centrale
  $q_i' = q_i + f_i \quad 0 \le i \le n$ (situazione nel vicinato dopo la distribuzione)

- **Vincoli:** $q_d = \Sigma_i f_i \quad 0 \le i \le n$
  $\Sigma_{i<j} \, | q_i' - q_j' | \quad 0 \le i < j \le n$ **deve essere minimizzata dai valori di** f_i

- **Algoritmo di minimizzazione delle differenze**
  (a) tutte le celle del vicinato sono etichettate "non escluse" dalla distribuzione:
  A è l'insieme delle celle "non escluse"
  (b) la "media di q" ($media\_q$) è trovata per l'insieme A delle celle non escluse:
  $media\_q = (q_d + \Sigma_i q_i) / \#A \quad i \in A$
  (c) ogni cella x con $q_x > media\_q$ viene **esclusa**
  (d) finché qualche cella viene esclusa, vai al passo (b).
  (e) calcolo dei flussi minimizzanti: $f_i = media\_q - q_i \quad i \in A$; $f_i = 0 \quad i \notin A$

Algoritmo di minimizzazione delle differenze (2)

Algoritmo di minimizzazione delle differenze

(a) Tutte le celle del vicinato sono etichettate "non escluse" dalla distribuzione:
A è l'insieme delle celle "non escluse"

(b) La "media di ***q***" (***media_q***) è trovata per l'insieme ***A*** delle celle non escluse:

$$media\_q = (q_d + \Sigma_i q_i)/\#A \quad i \in A$$

(c) La cella ***x*** con $q_x > media\_q$ è esclusa

(d) Finché qualche cella viene esclusa, vai al passo (b).

(e) calcolo dei flussi minimizzanti: $f_i = media\_q - q_i \quad i \in A$; $f_i = 0 \quad i \notin A$

ESEMPIO DI MINIMIZZAZIONE (AC bidimensionale con vicinanza di von Neumann):

	1	
3	0	2
	4	

$q_d = 9,\ q_0 = 81,\ q_1 = 100,\ q_2 = 76,\ q_3 = 83,\ q_4 = 71,$

	100				*				*				100	
83	81:9	76		83	81:9	76		*	*9	76		83	81	76:2
	71				71				71				71:7	
media_q=420/5=84 la cella 1 è esclusa				***media_q***=320/4=80 le celle 0 e 3 sono escluso				***media_q***=156/2=78 nessun'altra cella è esclusa				$f_2=2$ $f_4=7$ $f_0=f_1=f_3=0$		

Algoritmo di minimizzazione delle differenze (3)

- L'algoritmo di minimizzazione delle differenze calcola i flussi minimizzanti, ma raramente risulta applicabile in questa forma estremamente semplice.
- Alcune osservazioni:

velocità v: bisogna valutare se in un passo dell'AC il flusso minimizzante possa passare tutto da una cella all'altra; questo dipende da considerazioni sulla sua velocità v; fra l'altro deve valere $v \leq p_d / p_t$, altrimenti il flusso deborda dal vicinato della cella, ciò comporta un vincolo nella scelta dei parametri p_d e p_t.

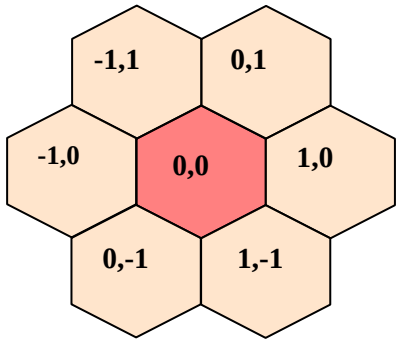
flusso effettivo: uno dei modi più semplici di tener conto della velocità v è quello di considerare che solo una parte del flusso minimizzante passa: sia d_{max} la distanza massima che deve essere percorsa affinché il flusso minimizzante f possa passare tutto, d_{eff} la distanza effettiva percorsa in relazione alla velocità v, allora la parte di flusso minimizzante che passerà effettivamente f_{eff} sarà $f_{eff} = f \cdot d_{eff} / d_{max}$.

potenziale attrattivo/repulsivo: si possono dare situazioni in cui bisogna specificare particolari condizioni di attrazione o repulsione, ad esempio differenze (gradienti) di temperatura e di cibo fra aree (fra celle dello spazio cellulare) possono determinare spostamenti di insetti, (che altrimenti sarebbero casuali e dipenderebbero dalla loro concentrazione) in tal caso "flussi" di insetti possono essere determinati dall'algoritmo di minimizzazione, definendo funzioni di potenziale attrattivo per temperatura e cibo.

SCIDDICA: *Simulation through Computational Innovative methods for the Detection of Debris flow path using Interactive Cellular Automata* (release S3-hex)

SCIDDICA = <R, X, Q, P, σ>

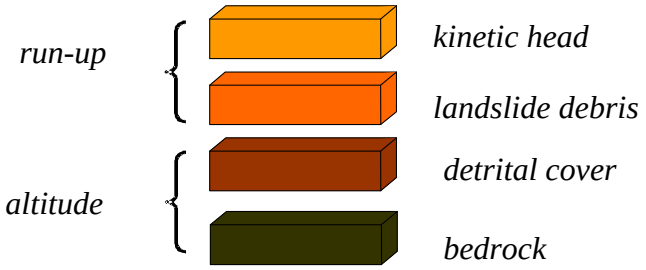
• **R** = {(x, y)| x, y ∈ ***N***, $0 \le x \le l_X$, $0 \le y \le l_Y$} is the set of points with integer co-ordinates in the finite region, where the phenomenon evolves. ***N*** is the set of natural numbers.



• **X** = {(0,0), (0,1), (0,1), (1,0), (-1,0), (-1,1), (1,-1)} is the set, which identifies the geometrical pattern of the cells, which influence the cell state change.

• The finite set **Q** of states of the ea: $\mathbf{Q = Q_a \times Q_{th} \times Q_r \times Q_d \times Q_o^6 \times Q_i^6}$

Q_a	altitude of the cell (related to bedrock and depth of detrital cover)
Q_{th}	thickness of landslide debris
Q_r	run-up height (it is a function of the kinetic energy of the landslide and expresses the height that can be overcome by the flowing debris)
Q_d	depth of detrital cover that can be eroded and hence transformed into landslide debris
Q_o (Q_i)	debris outflow (inflow)



SCIDDICA (release S3-hex)

• **P** is the set of *global parameters* of SCIDDICA $\mathbf{P = \{p_c, p_t, p_{adh}, p_t, p_r, p_{rl}, p_{mt}, p_{pef}\}}$

p_c	apothem of the hexagonal cell	1.5 m
p_t	temporal correspondence of a step of SCIDDICA	0.25 s (?)
p_{adh}	adhesion	0.001 m
p_f	elevation threshold for movement (related to friction angle)	0.1 m
p_r	relaxation rate of debris landslide outflows	1
p_{rl}	run-up loss	0.6 m
p_{mt}	activation threshold for mobilisation	2 m^2
p_{pef}	factor of progressive erosion	0.04

• $\sigma: \mathbf{Q^7 \rightarrow Q}$ is the deterministic transition function of the *CA*

elementary process in order	***input***	***variations/determinations***	***updating***
soil mobilisation	$Q_{th} \times Q_d \times Q_r$	$\Delta(Q_a, Q_{th}, Q_d, Q_r)$	Q_a, Q_{th}, Q_d, Q_r
outflow of debris	$(Q_a \times Q_{th})^7 \times Q_r$	Q_o^6	Q_o^6, Q_i^6
debris mixing	$Q_{th} \times Q_i^6 \times Q_o^6 \times Q_r^7$	Q_{th}, Q_r	Q_{th}, Q_r
energy loss	$Q_{th} \times Q_r$	Q_r	Q_r

SCIDDICA (release S3-hex) transition function σ (1)

Internal transformation MOBILISATION $\sigma_M: Q_{th} \times Q_d \times Q_r \rightarrow Q_a \times Q_{th} \times Q_d \times Q_r$

The erosion of the detrital mantle is allowed, in proportion to the energy of the moving mass in the cell. That is expressed by $Q_{th} * Q_r > p_{mt}$.

The depth of eroded material, at each step, depends on the parameter p_{pef} . It is the minimum value between *1)* the depth of available detrital cover Q_d and *2)* the product $p_{pef} \times Q_{th} \times Q_r$

Local interaction: DEBRIS OUTFLOWS $\sigma_{DO}: (Q_{th} \times Q_a)^7 \times Q_r \rightarrow Q_o^6$

A preliminary test is executed in order to account the friction effects, that prevent debris outflows, when the height difference between the two cells is insufficient; the condition is expressed by the formula $(q[0]+p[0]-q[i])<p_f$.

Minimisation algorithm q_d = quantity, that may be distributed, in the central cell = $Q_r - p_{adh}$

q_0 = inamovable quantity in the central cell = $Q_a + p_{adh}$

application q_i = quantity in the cell i $1 \leq i \leq 6 = Q_a + Q_{th}$

SCIDDICA (release S3-hex) transition function σ (2)

Local interaction: DEBRIS MIXING $\sigma_{DM}: Q_{th} \times Q_i^6 \times Q_o^6 \times Q_r^7 \rightarrow Q_{th} \times Q_r$

Debris mixing involves the determination for the central cell:

a) the remaining debris thickness (**rem_th**): $rem\_th = Q_{th}[0] - \Sigma_j Q_o[j] \quad 1 \leq j \leq 6$

b) new debris thickness (**new_th**): $new\_th = rem\_th + \Sigma_j Q_i[j] \quad 1 \leq j \leq 6$

c) The run-up determination is calculated as the average weight of Q_r, by considering both the remaining debris in the central cell and the inflows:

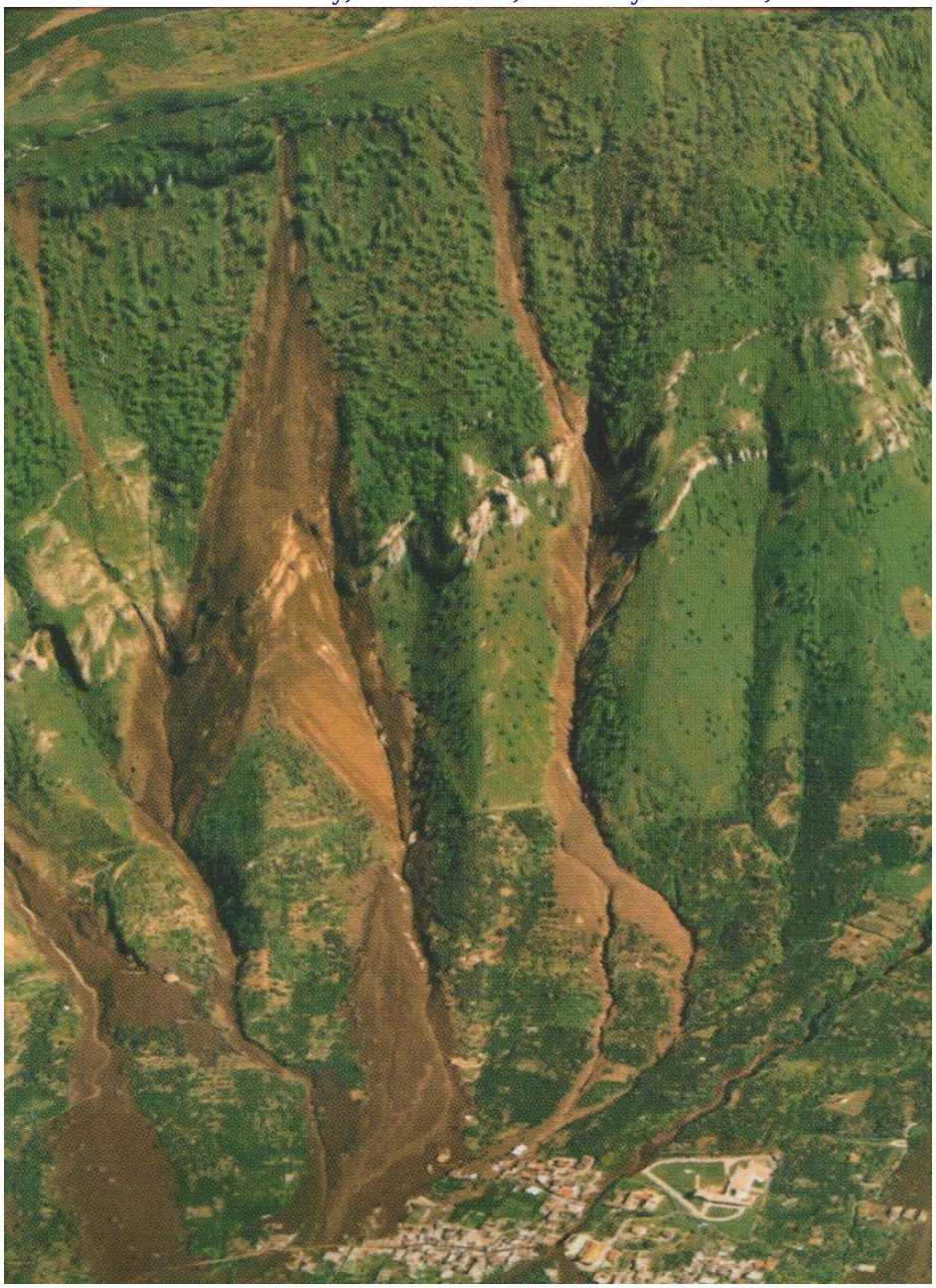
$$\left((Q_{th}[0] - \sum_{j=1}^{6} Q_o[j]) \times Q_r[0] + \sum_{j=1}^{6} (Q_i[j] \times Q_r[j]) \right) \Big/ \left(Q_{th}[0] + \sum_{j=1}^{6} (Q_i[j] - Q_o[j]) \right)$$

Internal transformation ENERGY LOSS $\sigma_{EL}: Q_r \times Q_{th} \rightarrow Q_r$

Energy loss , at each step, is computed in terms of run up loss and depends on the parameter $\mathbf{p_{rl}}$. It is the minimum value between *1)* the difference $\mathbf{Q_r - p_{rl}}$ and *2)* debris thickness $\mathbf{Q_{th}}$

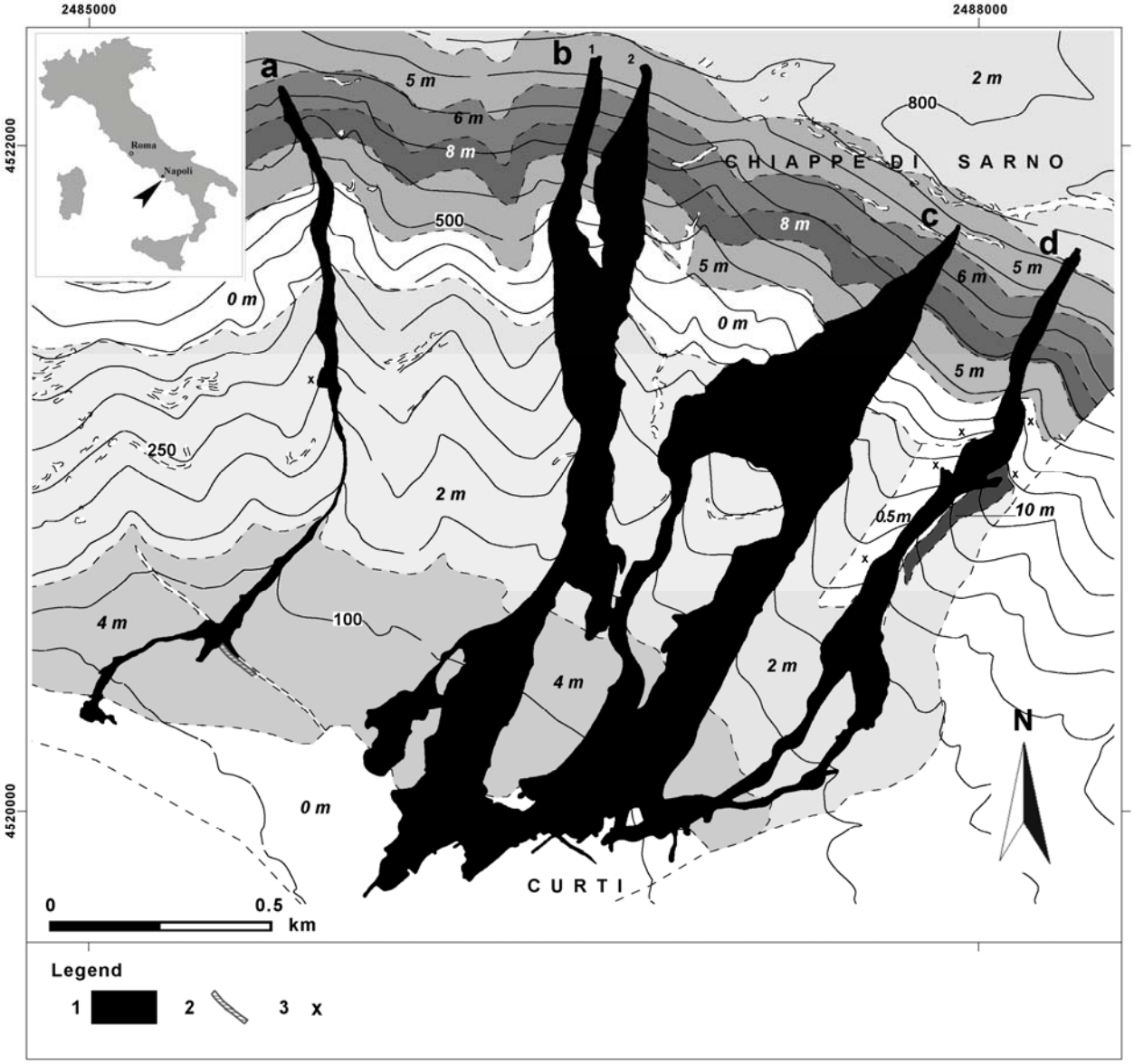
THE SARNO LANDSLIDES

On 5-6 May 1998, numerous rapid-extremely rapid debris flows were triggered by exceptional rainfalls in Campania, mostly on the slopes of Pizzo d'Alvano. Hundreds of small debris slides originated in the volcaniclastic mantle, and transformed into fast-moving debris flows. These latter generally eroded the entire depth of volcanic detrital cover, greatly increasing their initial volume. Landslides hit the urbanised areas, at the base of the slopes, killing 161 people and leaving more than 1,000 others homeless.



THE SARNO LANDSLIDES

A map of the disaster has been realised, at 1:5,000 scale: the geologic-geomorphologic context has been critically analysed, and the required information for simulation purposes acquired (e.g. pre-event conditions of slopes and detrital cover).



SCIDDICA(S3-hex) **application to the Sarno landslides** (Curti case)

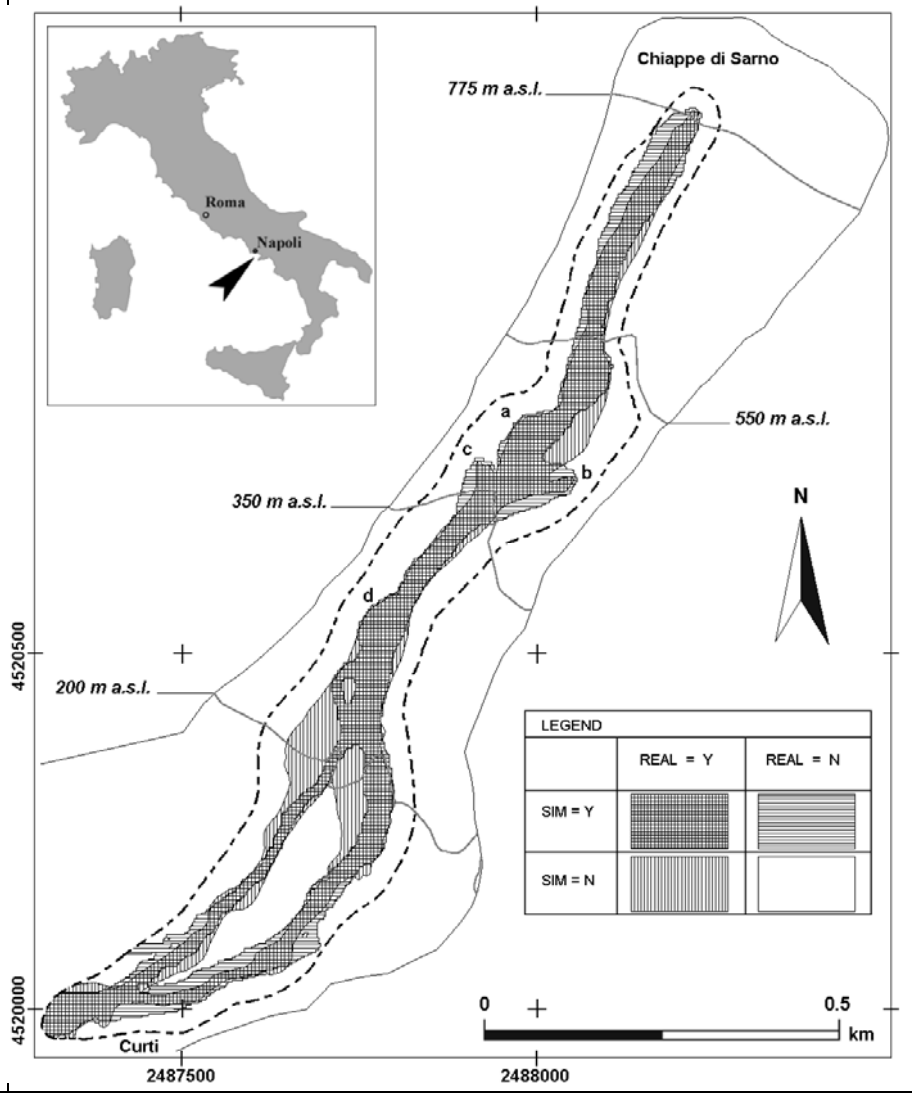
Simulation data: Real area (R) = m^2 96902

Simulation area (S) = m^2 91093

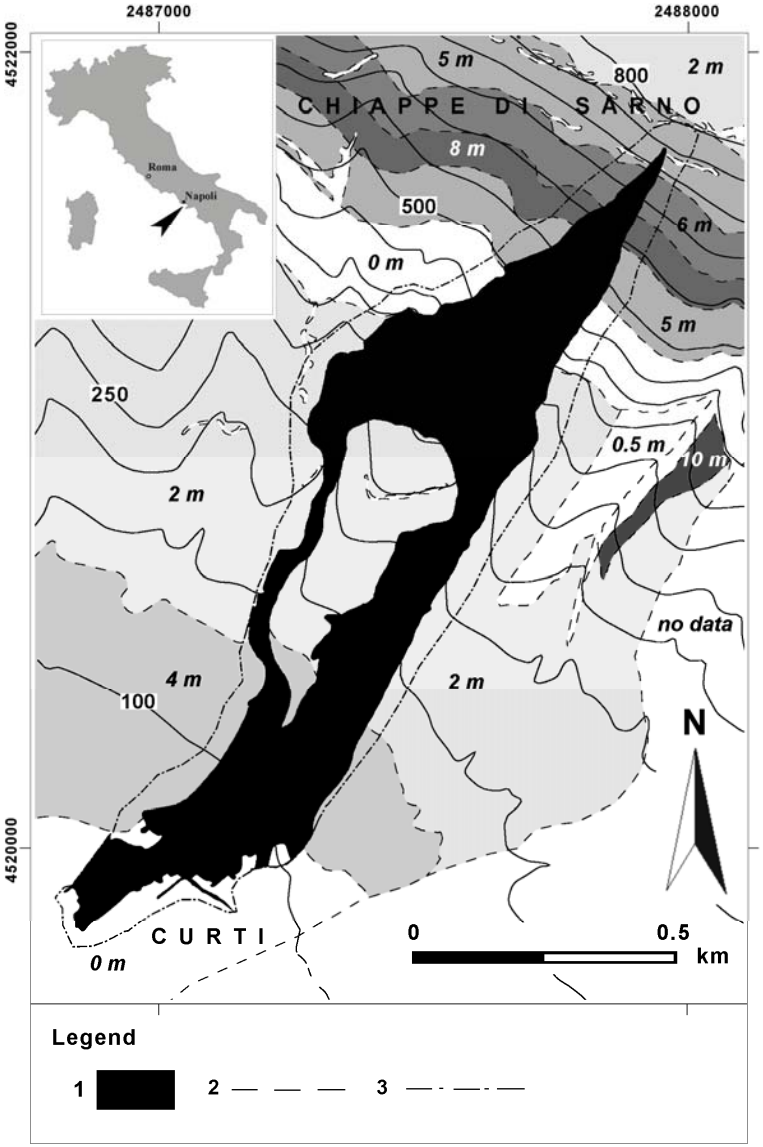
$R \cap S = m^2 70927$ - $R \cup S = m^2 117017$

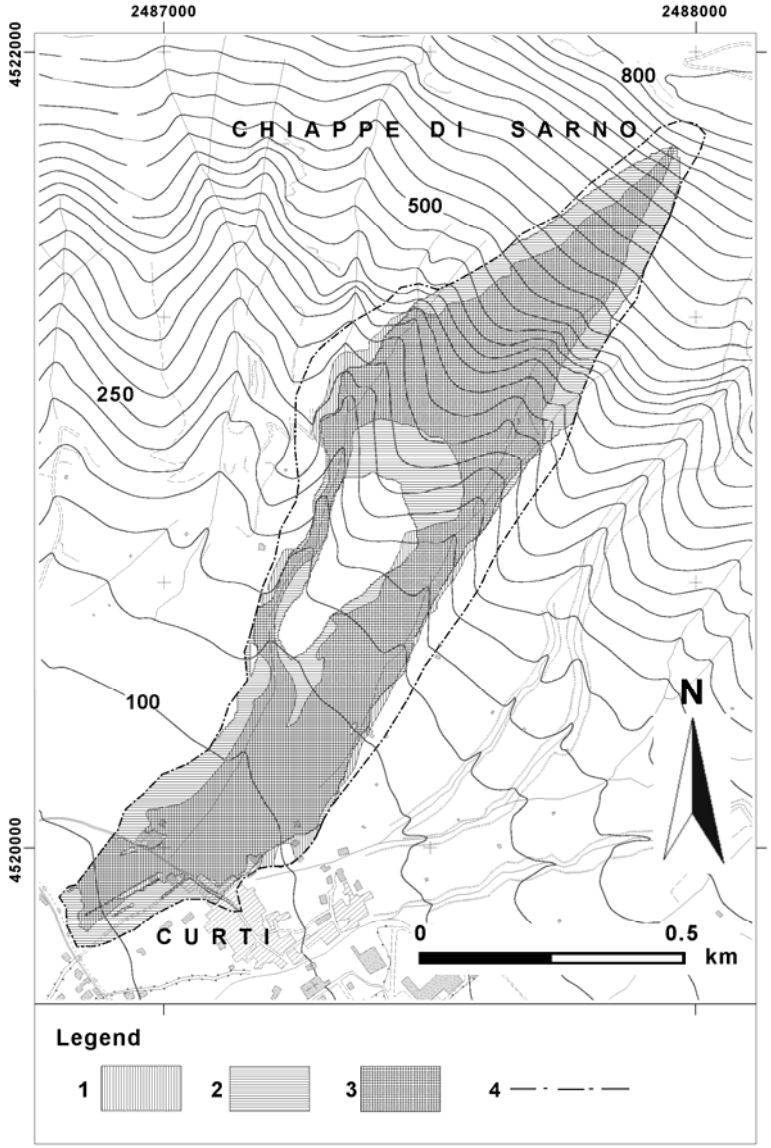
$(R \cap S)/R = 0.73$ - $(R \cap S)/S = 0.78$

$$\sqrt{((R \cap S)/(R \cup S))} = 0.78$$

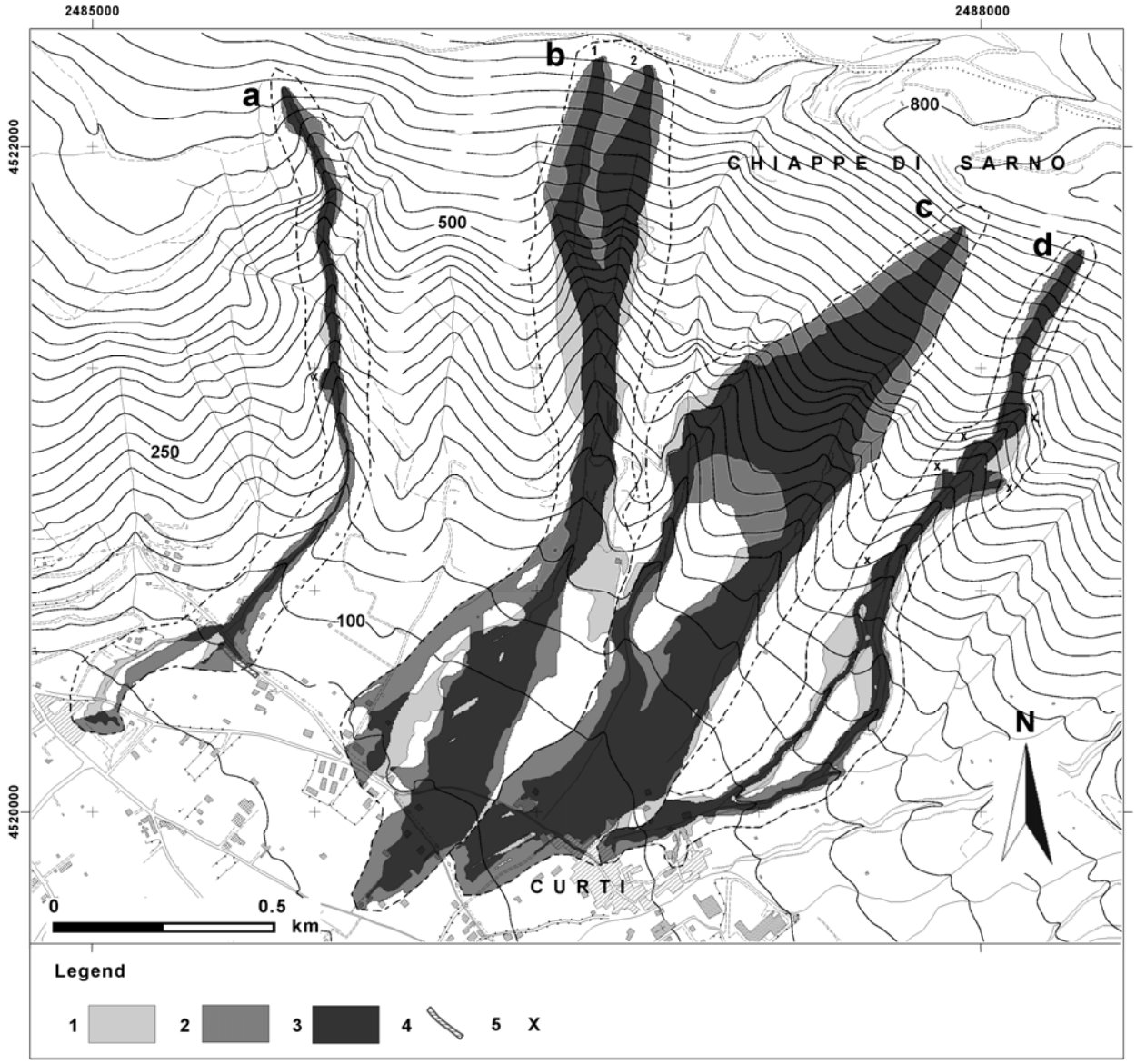


SCIDDICA(S3-hex) **application to the Sarno landslides** (Chiappe di Sarno case)

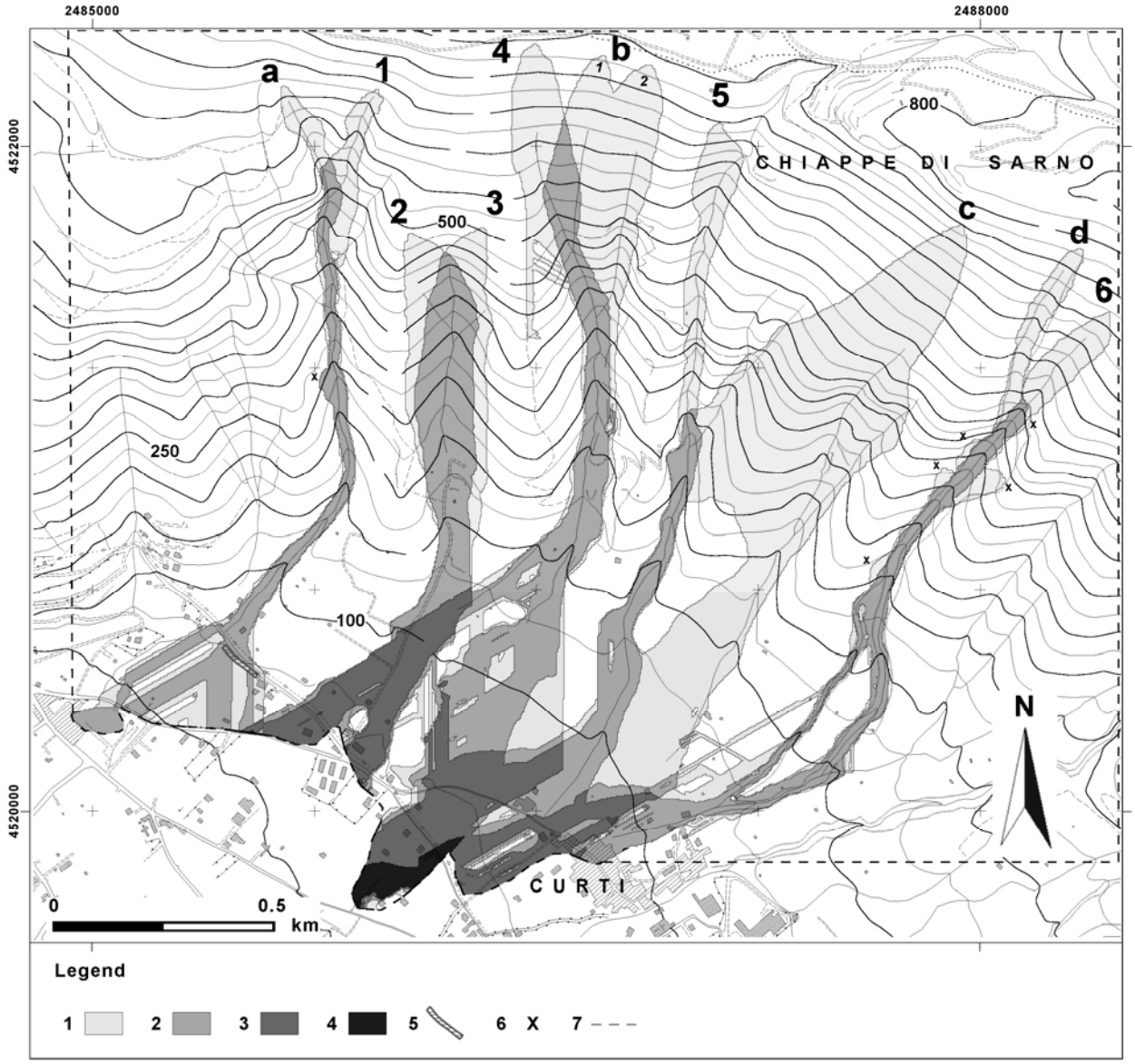




SCIDDICA(S3-hex) application to the Sarno landslides



SCIDDICA(S3-hex) debris-flow susceptibility assessment



COMMENTS AND CONCLUSIONS

- Cellular Automata may represent an alternative approach to differential equations in modelling complex systems, whose evolution is strongly dependent on local interactions of their constituent parts.

- The empirical method, here introduced, was successfully applied by the research group “Empedocles” to other macroscopic complex phenomena, such as soil contamination and bioremediation, forest fires, soil erosion by rain; new application fields are considered: pyroclastic flows, marine environment evolution.

- This empirical method permits to start with simple models, whose refinement can be performed in an incremental way, introducing other internal transformations and local interactions. This allows a careful monitoring of the model building phase by comparison between real phenomena and simulations.

- This empirical method involves, for each internal transformation or local interaction, the introduction of problem-specific parameters, whose determination may be performed by applying optimization methods to minimize the difference between model results and experimental data. Genetic algorithms were effective for applications with several parameters.

- It is important to define the limits to the application of the model to similar phenomena: e.g., SCIARA was validated for the Etnean lava flows in 1986/7 eruption and 1991/2 eruption. SCIARA application during the eruption in the 2001 Summer for the hazard analysis was possible, because Etnean lavas features don’t change significantly in the time. Cases, where the features change, involve a validation considering an interval of possible values of parameters, corresponding to different typologies of cases.

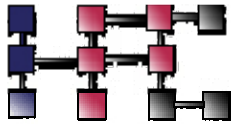
- This point is crucial; investigantions showed that there are different confidence intervals for phenomena of the same type.

- A last consideration can be added: the decomposition of the complex macroscopic phenomenon in internal transformations and local interactions seems to have encouraged interdisciplinary cooperations and exchange of information, at least in the cases treated here.

WIVACE 2008 – Venezia, 8-10 Settembre 2008
Workshop Italiano di Vita Artificiale e Computazione Evolutiva

A CELLULAR AUTOMATA MODEL FOR HIGHWAY TRAFFIC WITH PRELIMINARY RESULTS

Salvatore Di Gregorio[1,2], Renato Umeton[1,2]



[1]Dept. of Mathematics, [2]Center of High-Performance Computing,
University of Calabria, Arcavacata, 87036 Rende (CS), ITALY

Andrea Bicocchi[3], Andrea Evangelisti[3]

[3]Abstraqt srl, via Puccini 311, 55100 Lucca (LU), ITALY

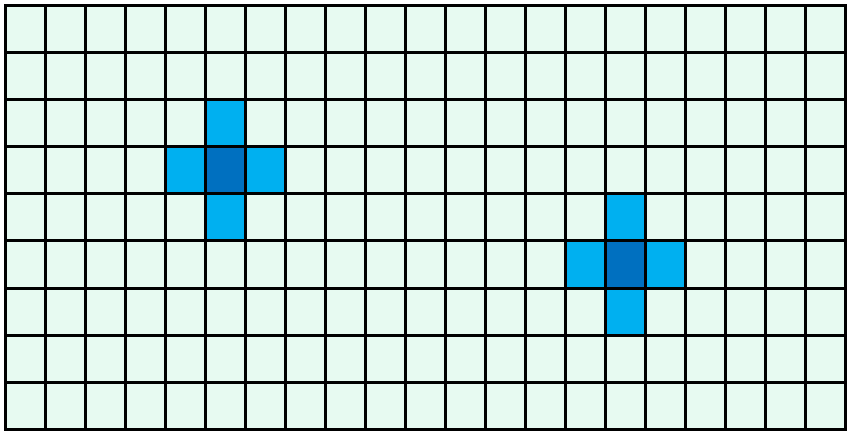
CONTENTS:

- Macroscopic Cellular Automata (CA) for modelling acentric complex macroscopic phenomena
- The STRATUNA general model :
  Simulation of highway TRAffic TUNed-up by cellular Automata
- The STRATUNA transition function
- STRATUNA first partial implementation: version $\beta 4$
- Results of simulations with STRATUNA$_{\beta 4}$
- Comments and conclusions.

Cellular Automata (CA)

- Intuitively a **CA** can be seen as a space (Z^d), partitioned in cells, each one embedding an identical input/output computing unit: a finite automata (the elementary automaton: **ea**)
- Each cell is characterised by its **state**. S is the finite set of the **ea** states
- Input for each cell is given by the states of m neighbouring cells, where the **neighbourhood** conditions are given by a pattern invariant in time & space.

Fragment of two-dimensional **CA** with von Neumann neighbourhood



- At time $t=0$, cells are in arbitrary states (initial conditions) and the **CA** evolves changing the state at discrete times, according to a same **ea transition function** $\tau: S^m \rightarrow S$, that is invariant in time & space.

CA and highway traffic

- **CA** may model highway traffic because of local and parallel characteristics of such a phenomenon: when highway structural features are fixed, the traffic evolution emerges by the mutual influences among vehicles in driver sight range.
- The main **CA** models of highway traffic may be considered “simple” in terms of external stimuli to the driver and corresponding reactions, but they are able to reproduce the basic different phases of traffic flow: free flow, wide moving jams and synchronized flow.
- Our approach use “**Macroscopic” CA** for a more accurate modelling: e.g., the vehicle space location is not identified by a sequence of full cell as in other CA models, but it permits to consider portions of cell and positions between two lanes, occupied by a vehicle.

CA for modelling complex macroscopic phenomena

- The abstract **CA** must be related univocally to the real macroscopic phenomenon in time and space : the **cell** corresponds usually to a portion of the space; so the cellular space must be three-dimensional
- **Global parameters** must be considered: at least the size of the cell p_s and the time corresponding to the **CA** transition step p_t,
- The state is composed by **substates**, each substate describes a feature of the space portion related to the own cell, e.g. the substate *temperature*;
- The substate value is considered always **constant** inside the cell
- **two** dimensions (**one** dimension) suffice, if quantities concerning the third (and second dimension) may be considered substates (e.g., *altitude*, *curvature radius*).
- The transition function τ is split into **"elementary" processes**, $\tau_1,\tau_2,..$; each elementary process involves the update of the state of each cell.
- Sometimes, an **input** from the "**external world**" to some cells of the **CA** must be considered; it accounts for external influence which cannot be described in terms of local rules (e.g. vehicle generation at tollgates) or for some kind of probabilistic approach to the phenomenon.
- Of course **special and/or additional functions** must be given for that type of cells.

The STRATUNA general model

STRATUNA = $< R, E, X, P, S, \mu, \gamma, \tau >$

- $R=\{x \mid x \in Z, 1 \le x \le n\}$
  is the set of ***n*** cells, forming the highway.
- $E \subset R$ is the set of entrance-exit cells in ***R***, where vehicles are generated and annihilated.
- $X = <-b, -b+1, \ldots, 0, 1, \ldots, f>$ defines the **ea** neighbouring, i.e the forward (***f***) cells and backward (***b***) cells in the driver sight, when visibility is maximum (no cloud, sunlight etc.).
- $P = \{length, width, clock, lanes\}$ global parameters
  - ***clock*** is the **CA** time step (driver reaction time (0.5s ÷1s)
  - ***length*** is the cell length (m.5)
  - ***lanes*** is the number of highway lanes (1, 2, ... from right to left) more lane 0, the entrance/exit/emergency lane.
  - ***width*** is the cell width (m.x × # ***lanes***)

STRATUNA = $<R, E, X, P, S, \mu, \gamma, \tau>$

- $S = Static \times Dynamic \times (Vehicle \times Driver)^{lanes}$ specifies the high level **ea** substates:
  - statistical and dynamical features of highway segment corresponding to the cell
  - vehicle and driver pair features for each lane
- $\mu: Z \times R \rightarrow Dynamic$ is the weather evolution function for ***Dynamic*** values at each ***clock*** for each cell $c \in R$.
- $\gamma: Z \times E \rightarrow Vehicle \times Driver$ is the vehicle-driver pair generation function at each ***clock*** for each cell $c \in E$.
- $\tau: S^{b+1+f} \rightarrow S$ is the **ea** transition function (the real visibility could be reduced to ***b'*** backward cells and to ***f'*** forward cells by function μ) .

$S = Static \times Dynamic \times (Vehicle \times Driver)^{lanes}$

- ***Static***
  - ***CellNO, Slope, CurvatureRadius, SurfaceType, SpeedLimit, Lane1SpeedLimit***
- ***Dynamic***
  - ***BackwardVisibility, ForwardVisibility, Temperature, SurfaceWetness, WindDirection,WindSpeed***
- ***Driver***
  - ***Origin, Destination, DesiredSpeed,***
  - ***PerceptionLevel*** concerns the perception of the objectively safe zones (their widths are % reduced or slightly increased),
  - ***Reactivity*** is a collection of constants for determining costs of possible actions,
  - ***Aggressiveness*** forces the deadlocks, that could be generated by a cautious ***PerceptionLevel***

- **Vehicle**
  - **Type** *motorcycle, car, bus/lorries/vans, semitrailers/articulated* ,
  - **Length, MaxSpeed, MaxAcceleration, MaxDeceleration, CurrentSpeed, CurrentAcceleration, Xposition, Yposition,**
  - **Indicator** with values: *null, left, right, hazard lights* ,
  - **StopLights,** with values: *on, off*
  - **WarningSignal** with values: *on, off*

    demand (e.g. sounding the horn, blinking high-beam lights, etc.) to free the lane immediately ahead of own vehicle.

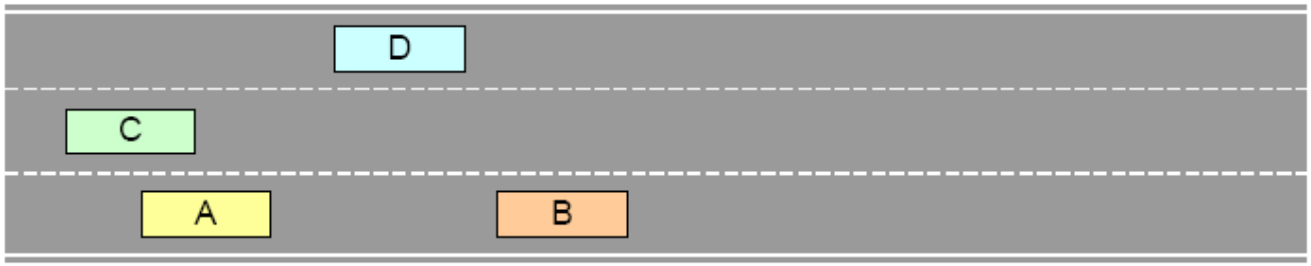
The STRATUNA transition function τ

- Weather & highway condition influence:
  - ***surface_slipperiness*** is computed by ***SurfaceType***, ***SurfaceWetness***, ***Temperature***
  - ***max_acceleration , max_deceleration*** are computed by ***surface_slipperiness***, ***Slope, CurvatureRadius, WindDirection,WindSpeed, MaxSpeed, MaxAcceleration, MaxDeceleration,***
  - ***max_speed*** is computed by ***SpeedLimit, Lane1SpeedLimit, BackwardVisibility, ForwardVisibility, max_acceleration***, ***max_deceleration***
  - *if* (***max_speed*** *<* ***DesiredSpeed*** *)*
    desired _speed *=* ***max _speed*** *;*
    else ***desired _speed*** *=* ***DesiredSpeed****;*

The STRATUNA transition function τ

Computation of the "free zones" for vehicle *V* :

It determines "**objectively**" all the zones in the different lanes, that cannot be occupied by the vehicles around *V*, considering the range of the speed potential variations and the lane change possibility, that is always signalled by Indicator.



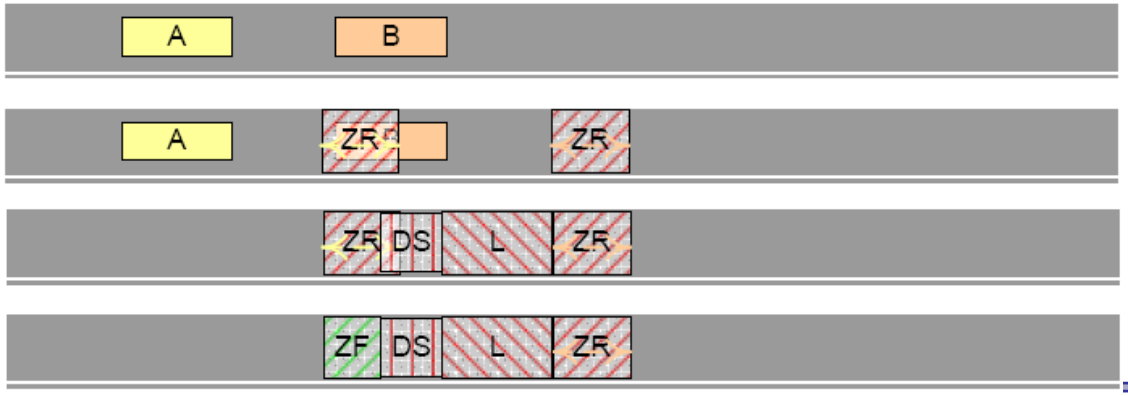
ZR is the interval of position, which the vehicle front can assume in the next step, considering ***max_acceleration*** and ***max_deceleration***

L is the vehicle length and **DS** is the distance of security; **IZ=DS+ZR+L** constitute the zone, involved by the vehicle (A) at the next step

The STRATUNA transition function τ

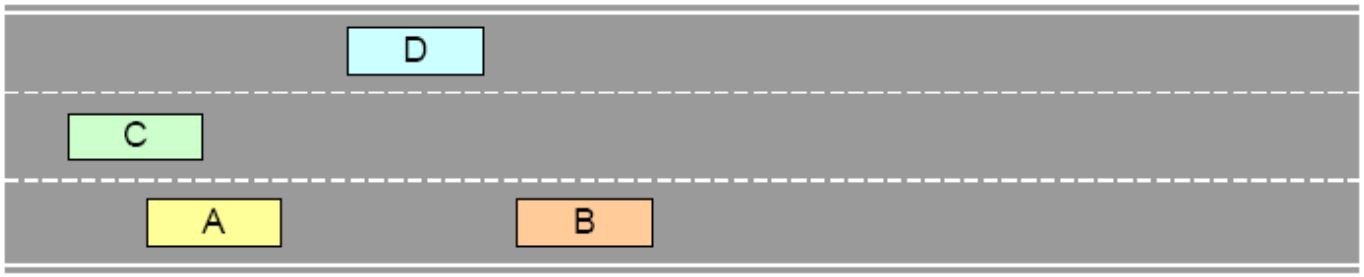
- Computation of the "free zones" for vehicle *V*:

- The "free zone" for A is given by $\mathbf{ZF_A = ZR_A \cap (\neg IZ_B)}$ if only the vehicle B is in the neighbouring.
- Two examples: null intersection for the above figure (well spaced vehicles in relation to speed), reduced $\mathbf{ZF_A}$ in the fig. below.

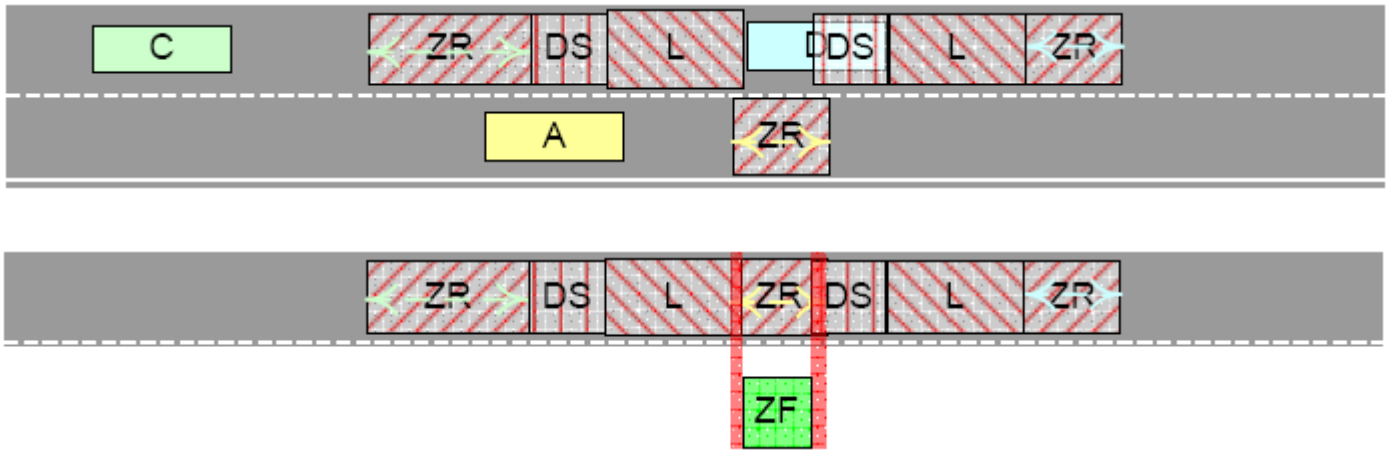


The STRATUNA transition function τ

- Computation of the "free zones" for vehicle *V*:



- Vehicle A could be interested to change lane, A must account for $\mathbf{IZ_C}$ and $\mathbf{IZ_D}$ (D is in the third lane, but it could change lane), consequently the free zone of A in left lane must be computed.
- The free zone of A is only in the current lane of A; another example is shown in the figure below with D shifted.

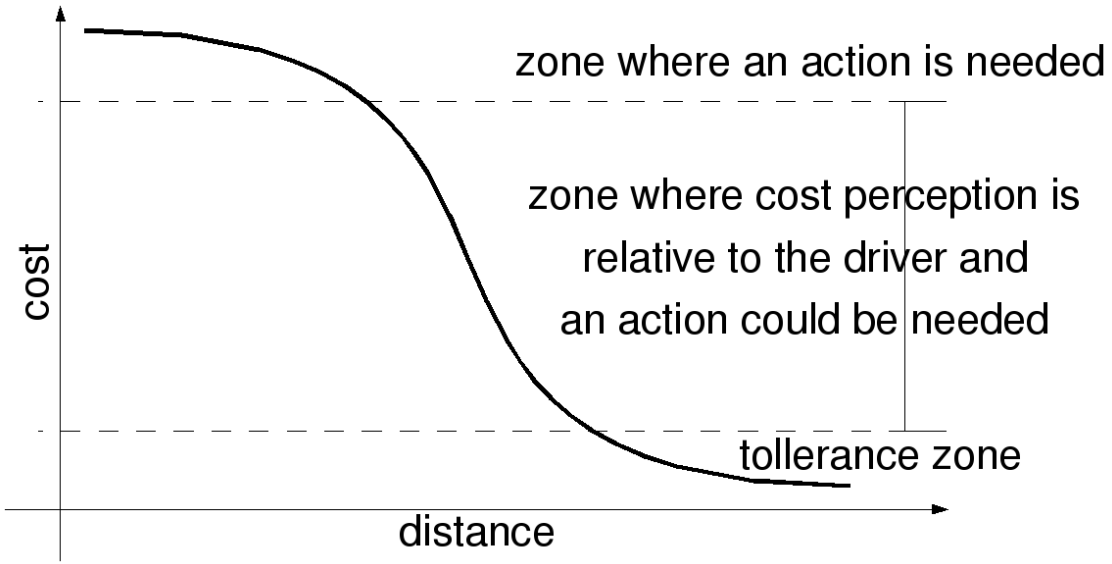


The STRATUNA transition function τ

The driver "subjectivity" :

The driver aims in the other cases to reach/maintain the desired speed, different options are perceived available, each one is constituted by actions involving costs (e.g. the cost of the gap between the new value of ***CurrentSpeed*** and the ***desired_speed***). The driver chooses the option among all the possible ones, with minimal sum of the costs.

Example of function that connects the distance from front vehicle with a cost.



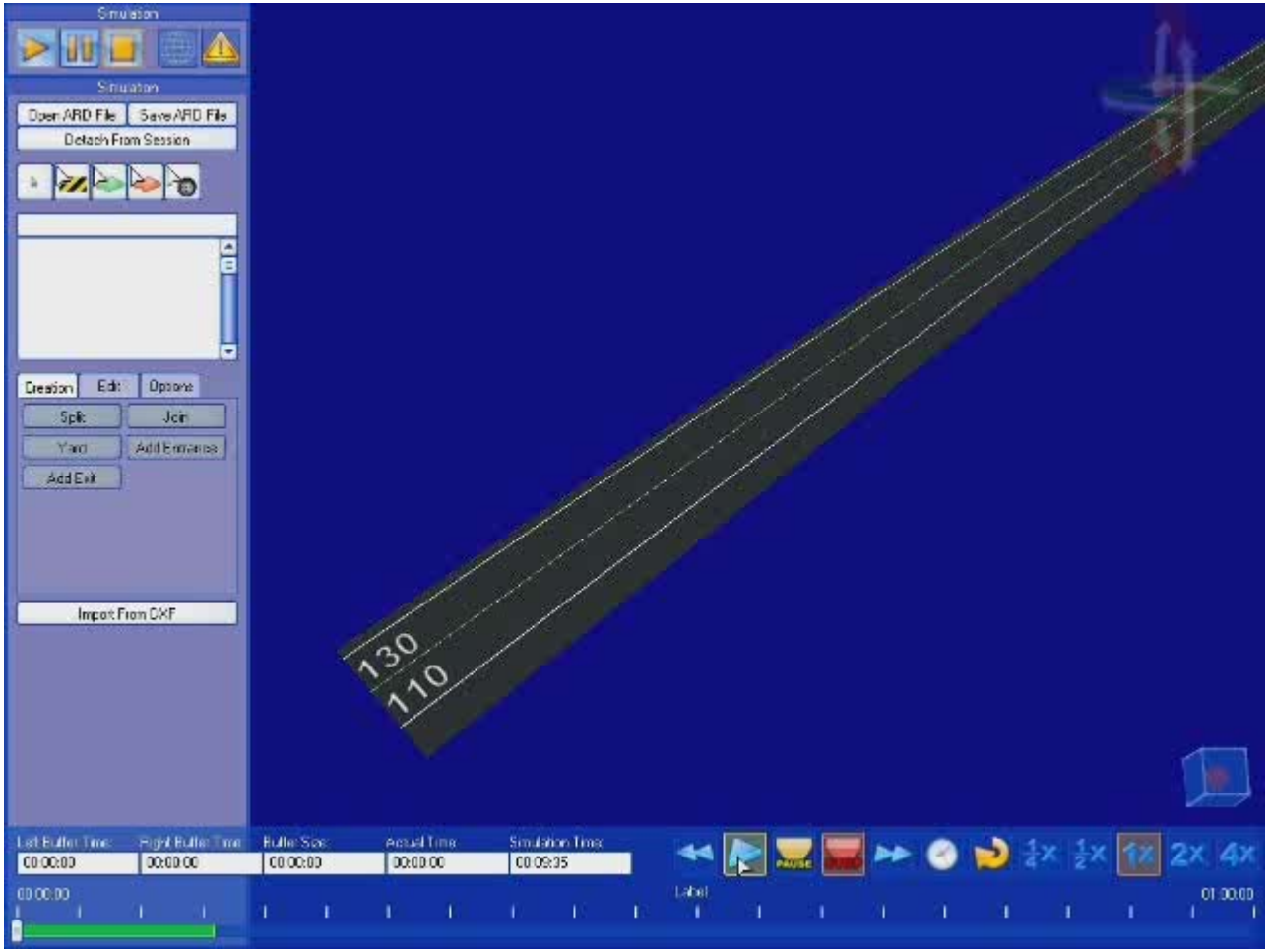
STRATUNA first partial implementation

- **$STRATUNA_{\beta 4} = \langle R, E, X', P, S', \gamma_{\beta 4}, \tau_{\beta 4} \rangle$**
- The function μ disappeared, because no weather evolution is considered, but only constant average conditions. ***Dynamic*** substate is no more considered.
- ***X'*** **= <-*r*,-*r* +1, … , 0, 1… , *r*>** substitutes ***X*** where ***r*** is a radius, accounting for the average visibility of an average driver.
- ***Indicator*** lacks of hazard lights value, ***PerceptionLevel*** value is always 1,
- Behaviour involving ***Aggressiveness*** was not implemented and ***Reactivity*** is considered only for "staying far from desired speed".

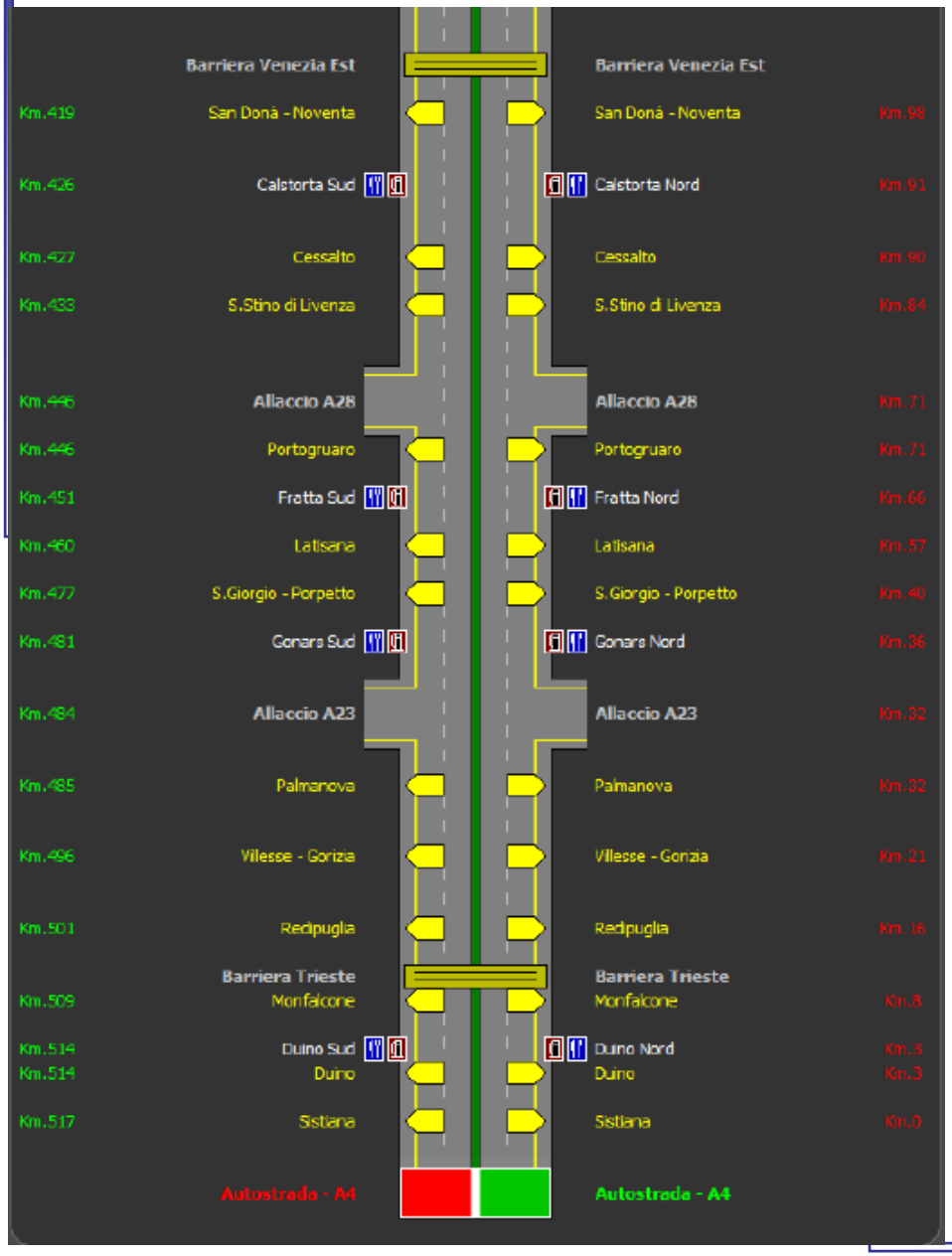
Outline of β4 transition function


```
BEGIN: TransitionFunction()
FindNeighbours(); ComputeSpeedLimits();
ComputeTargetSpeed(); DefineFreeZones();
AssignTheCost_PM_WhereAFreeZoneIsReduced();
if(ManouvreInProgress())
   continueTheManouvre(); return;
if(myLane==0) //I'm on a ramp
   if(IWantToGetIn())
      if(TheRampEnded())
         if(ICanEnter())
            enter(); return;
         else
            if(IHaveSpaceProblemsForward())
               slowDown(); return;
            else followTheQueue(); return;
      else //the ramp is not ended yet
         if(IHaveSpaceProblemsForward())
            followTheQueue(); return;
         else keepConstantSpeed(); return;
   else //I want to get out
      if(TheRampEnded()) deleteVehicle(); return;
      else
         if(IHaveSpaceProblemsForward())
            followTheQueue(); return;
         else keepConstantSpeed(); return;
   //end lane==0
else if(myLane==1)
   if(MyDestinationIsNear()) slowDown();
   if(MyDestinationIsHere()) goInLowerLane();
else //myLane==2 or more
   if(ICanGoInLowerLane())
      if(GoingInLowerLaneIsForcedOrConvenient())
         goInLowerLane();
   else //I cannot go in lower lane
      if(MyDestinationIsNear())
         slowDown(); goInLowerLane();
if(!IHaveSpaceProblemsForward()) //every lane
   if(TakeoverIsPossibleAndMyDestinationIsFar())
      if(TakeOverIsDesired()) takeover();
      else followTheQueue();
   else followTheQueue();
else //I have space problems forward
   if(TheTakeoverIsForced()) takeover();
return;
END;
```

Demonstration of STRATUNA , version β4

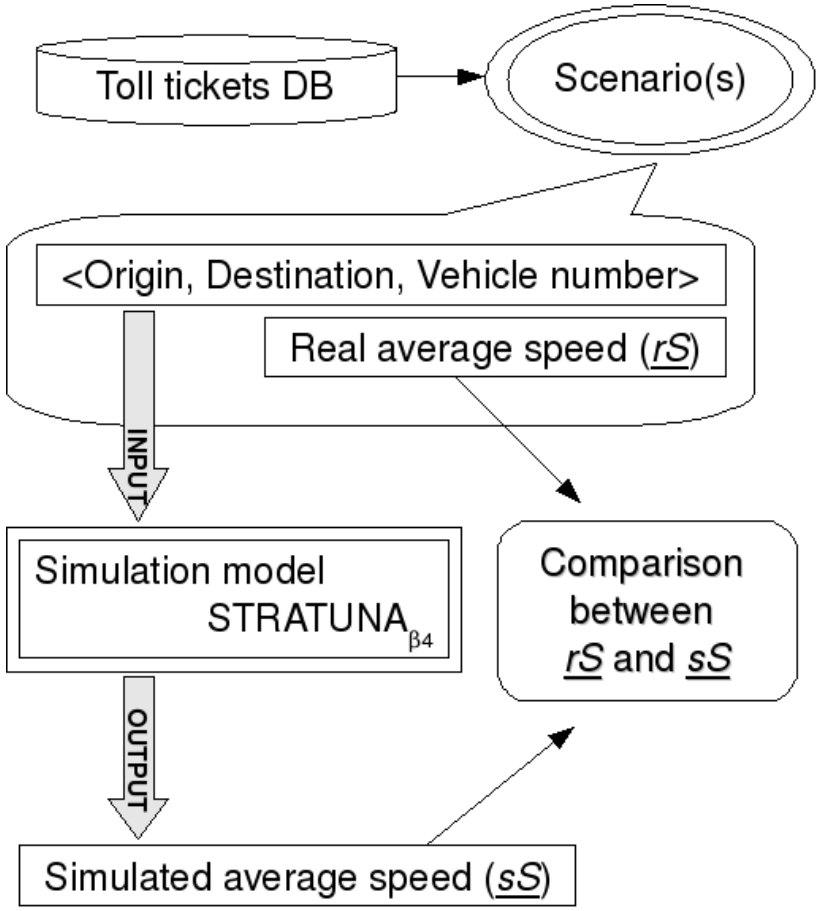


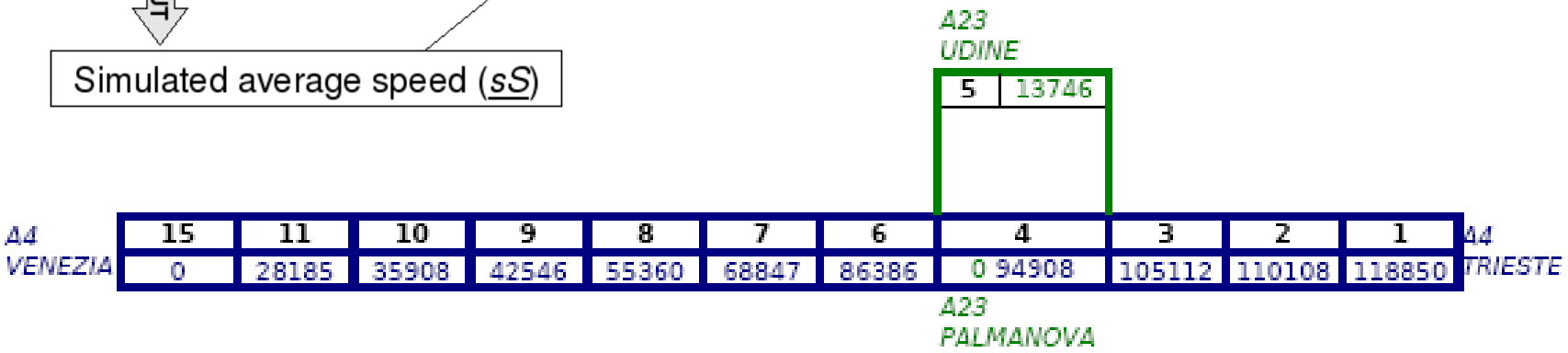
STRATUNA- β4 first application



- The generation function $\gamma_{\beta 4}$, was tailored for the traffic of Italian highway A4 Venezia-est/Lisert, characterized by two lanes and twelve entrances/exits.
- Data are composed by around 1 milion of tolltickets, they are related to 5 non-contiguous weeks and grouped in five categories, depending on vehicle number of axles (it is reducible to our vehicle classication).

STRATUNA- β4 first application



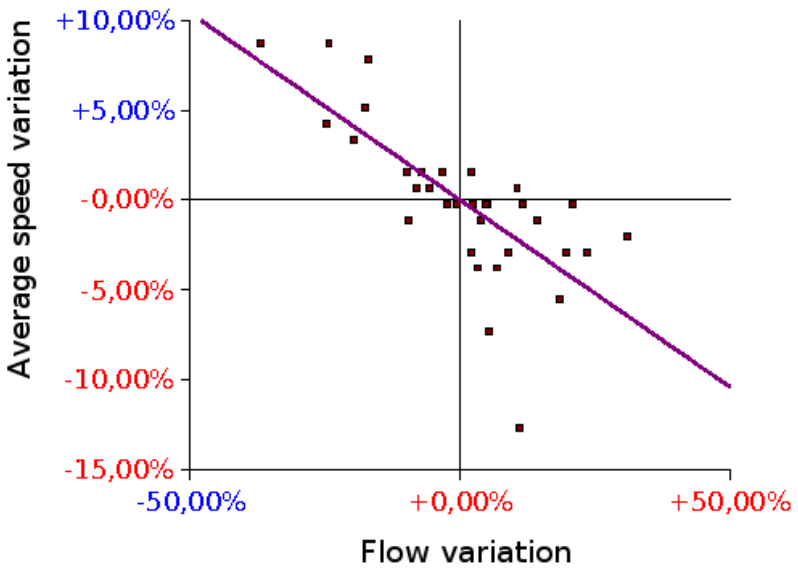


Data analysis for the Venezia-est/Lisert highway

Datasets include:

origin, destination,
vehicle type,
entrance and exit times.

Desired speed for each type of vehicle is deduced by highway data in free flow conditions for vehicles covering short distance in highway.

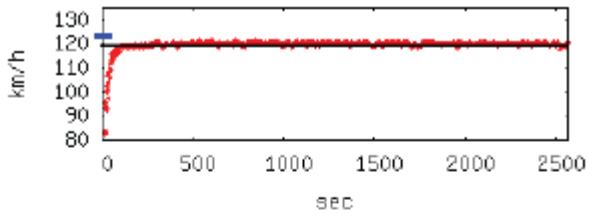


Vehicle type	Desired speed	Flow share
I	122.80 km/h	93.4%
II	112.77 km/h	4.6%
III	113.29 km/h	0.5%
IV	102.61 km/h	0.1%
V	93.90 km/h	1.4%

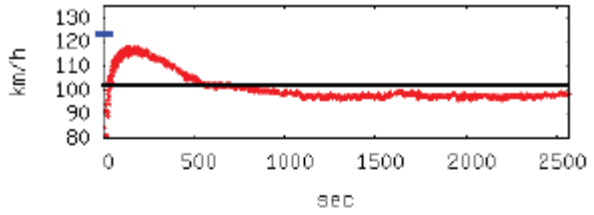
Some general statistics from the study of cleaned datasets is here summarized: each day is represented as a dot; a shift over x-axis and y-axis is a variation respectively of “total flow” and “average speed” from their averaged values over all the days

Average speed fluctuation in selected cases study

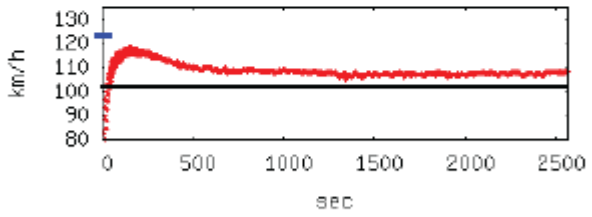
rS average real speed of vehicles of validation set (straight line),
sS step-by-step average simulated speed (fluctuating lines),
sDs average simulated Desired Speed (notch)
Average relative error between ***sS*** and ***rS***: e_1 (over all AC steps)
e_2 (calculated after 500 seconds)



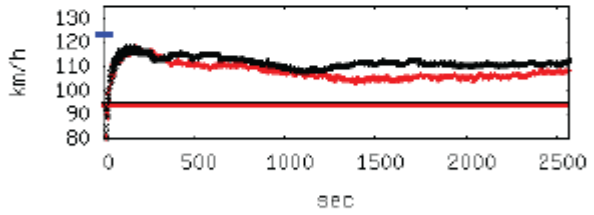
(a) Freeflow: *clock*=1s; err_1=1.29%; err_2=0.86%.



(b) Moderated flow: *clock*=1s; err_1=4.87%; err_2=3.44%.



(c) Moderated flow (tuned reactivity value): *clock*=0.75s; err_1=6.47%; err_2=5.83%.



(d) Locally congested situations. Lighter (darker) curve: *clock*=1s; err_1=14.79% (17.27%); err_2=13.94% (17.12%).

Conclusions

- Preliminary results of the reduced version β**4** of the **STRATUNA** model are very promising, considering that discrepancies between statistics deduced by real data and simulations are in part justified by unavoidable inaccuracies in the available real data and by imprecision introduced by parking in the rest and services areas.
- This is an interesting starting point in order to implement the full model. An important problem will be to tune some values of variables concerning the driver subjective behaviour to solve problems of congested situations.
- Such values cannot be deduced directly by data of real events as the ***DesiredSpeed***, therefore a calibration action must be performed.
- We intend to use the powerful tool of **Genetic Algorithms** in order to solve this crucial problem.
- Accessing to other types of data concerning highway traffic would be important for the approach completeness.

REFERENCES

1. S. Di Gregorio and R. Serra, *Fgcs* **16**, 259 (1999).
2. A. Schadschneider, *Phys. A* **372**, 142 (2006).
3. K. Nagel and M. Schreckenberg, *J. Phys. I* **2**, p. 2221 (1992).
4. D. E. Wolf, *Phys. A* **263**, 438 (1999).
5. W. Knospe, L. Santen, A. Schadschneider and M. Schreckenberg, *J. Phys. A: Math. Gen.* **33**, 477 (2000).
6. M. Lárraga, J. del Ríob and A. Icaza L., *Transport. Res. C.* **13**, 63 (2005).
7. S. Di Gregorio and D. C. Festa, *Cellular Automata for Freeway Traffic*, in *Proceedings of the First International Conference Applied Modelling and Simulation*, ed. G. Mesnard 1981, pp. 133–136.
8. S. Di Gregorio, D. Festa, R. Rongo, W. Spataro, G. Spezzano and D. Talia, *A microscopic freeway traffic simulator on a highly parallel system*, Advances in Parallel Computing Vol. 11 1996, pp. 69–76.

THANKS FOR YOUR ATTENTION