САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

Templet Parallel Computing System: Specification, Implementation, Applications

Sergei Vostokin
Professor, Information Systems and Technologies Department,
Samara University

ITNT-2017, Samara, 25-27 April, 2017

The Temple system aims to reduce the parallel programming complexity to the level of sequential programming for ones who develop their own HPC applications

Programming in Templet means:

- a view of parallel algorithm

  sequential algorithm + specification of parallelism

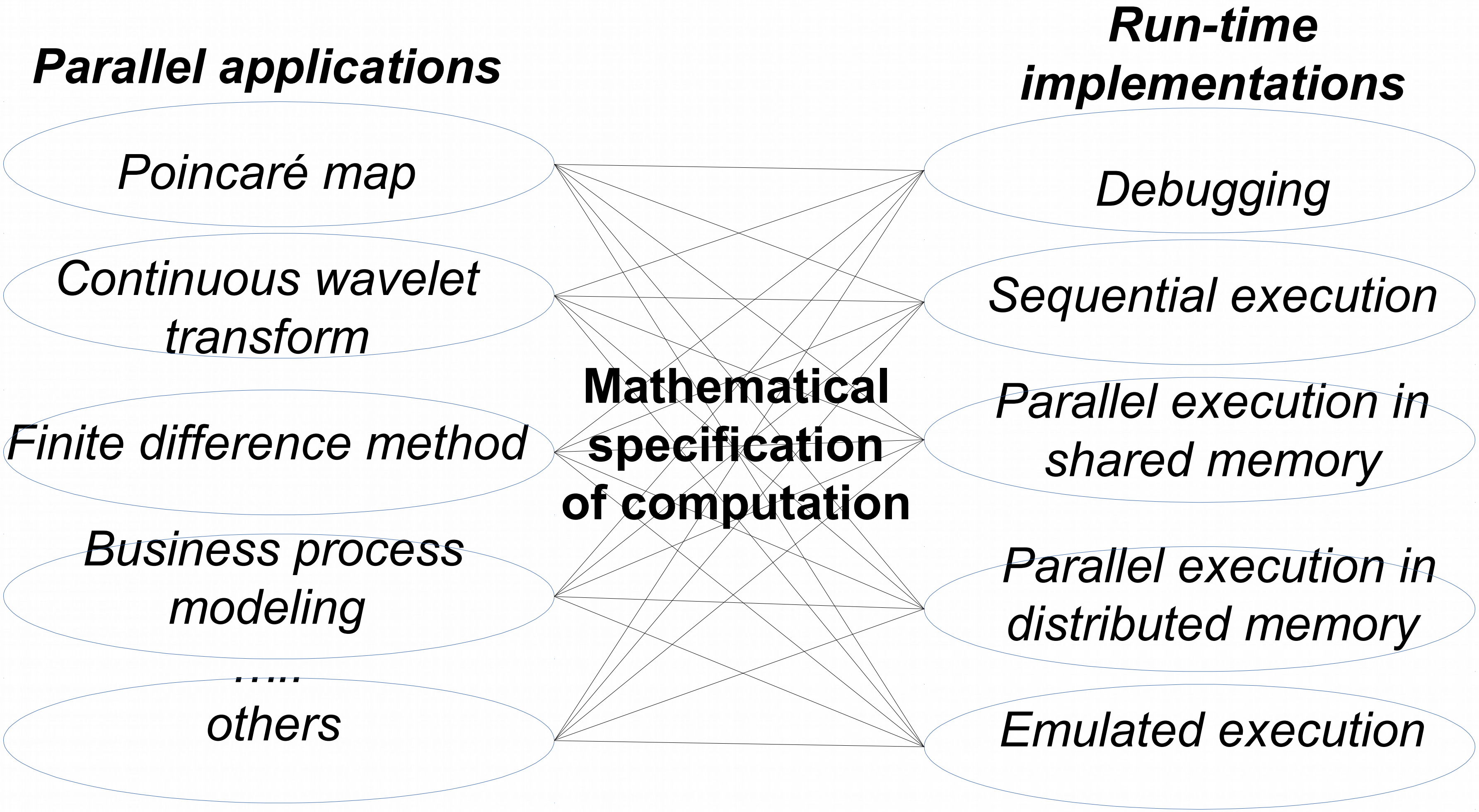
  → ***parallel algorithm***

- using standard language (C++), libraries (OpenMP, MPI), IDE, and the Templet *lightweight tooling* in application development

Research methods:

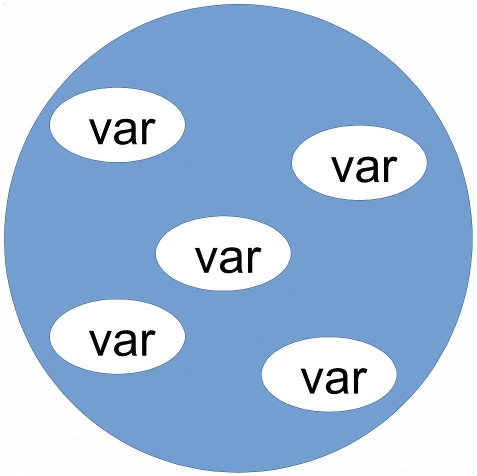
- actor model of execution (Carl Hewitt)
- temporal logic of actions (Leslie Lamport)
- algorithmic skeletons (Murray Cole)
- language-oriented programming (Martin Ward)

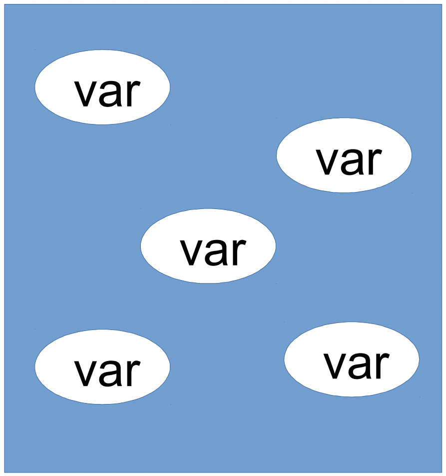
IT IS IMPORTANT TO UNDERSTAND HOW AN APPLICATION INTERACTS WITH ITS EXECUTION ENVIRONMENT



34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

$$F : Var \rightarrow \{actor, message\} \times N$$



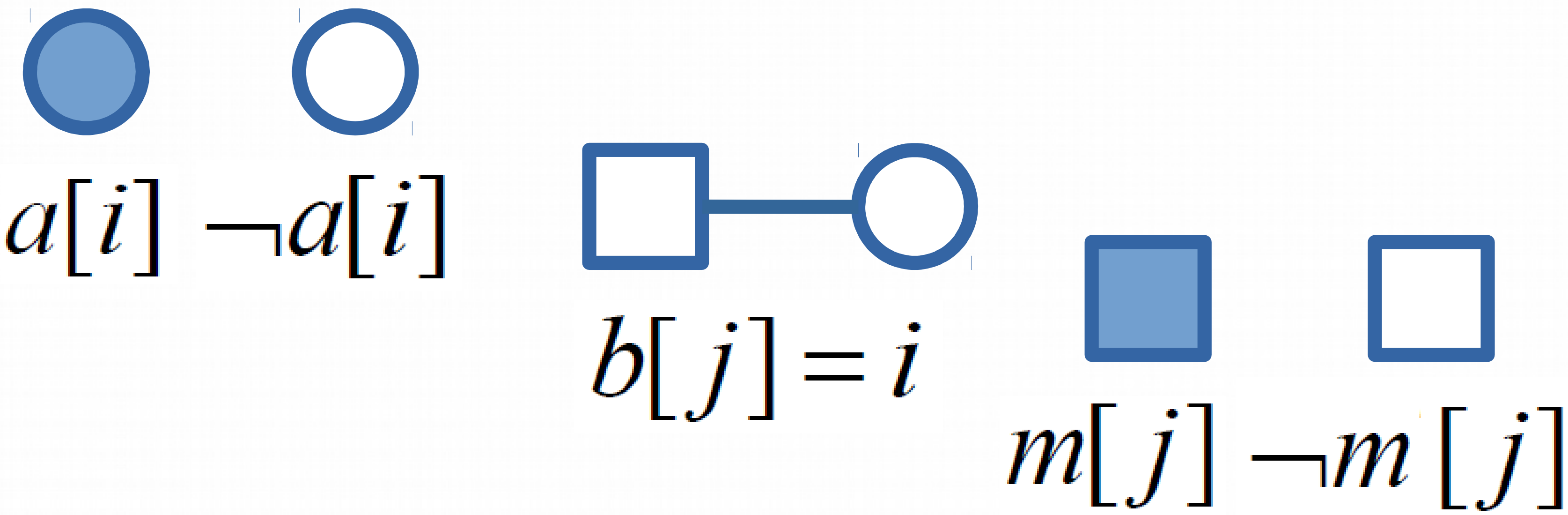


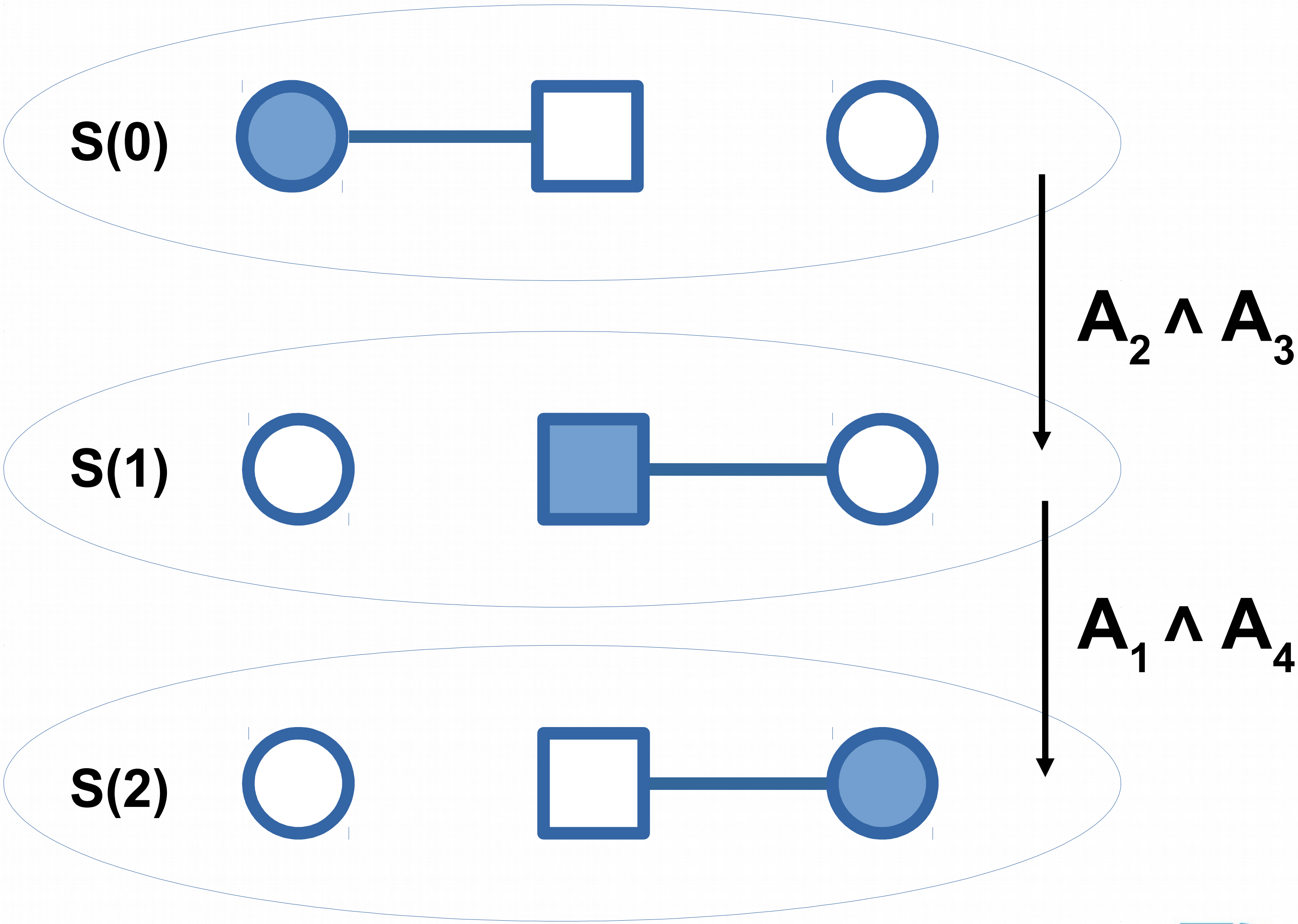
an actor

a message

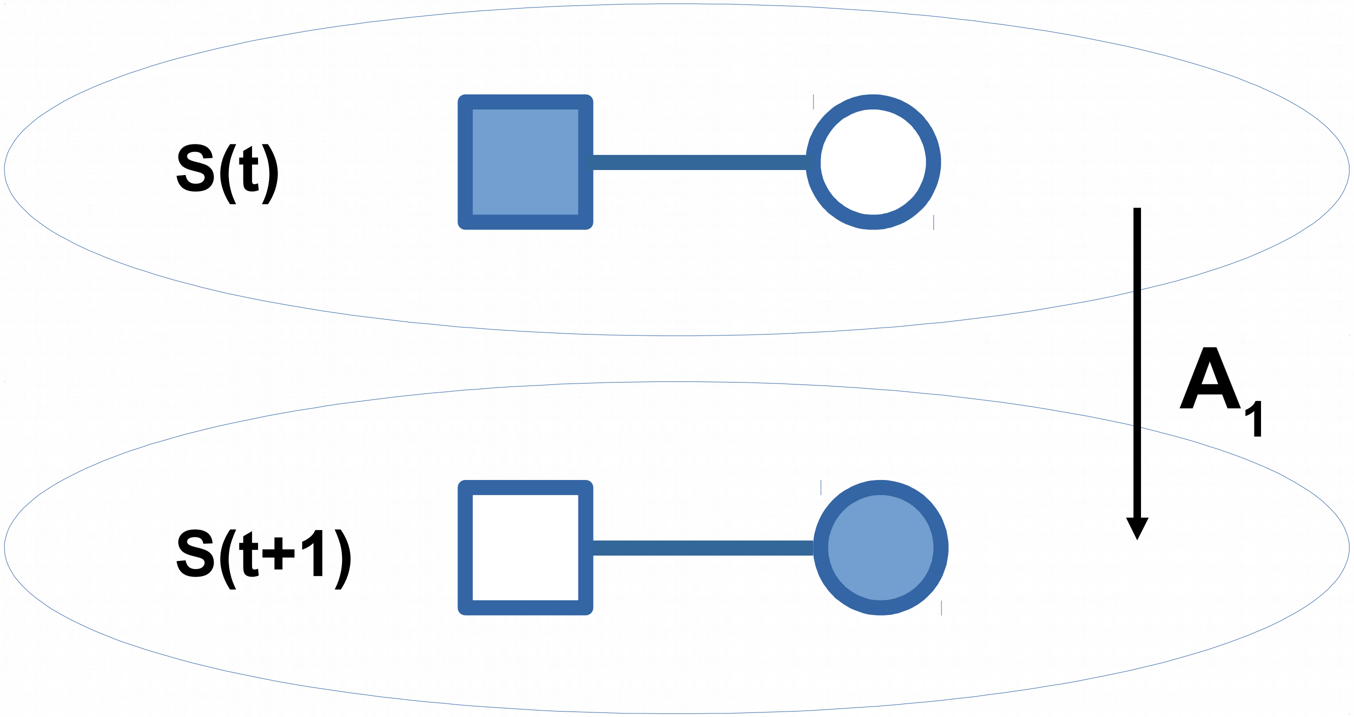
ACTOR SYSTEM RUN-TIME VARIABLES

$$\{a[1], a[2], \ldots, a[i], \ldots,$$
$$b[1], b[2], \ldots, b[j], \ldots,$$
$$m[1], m[2], \ldots, m[j], \ldots\}$$




S(0)
S(1)
S(2)
$A_2 \wedge A_3$
$A_1 \wedge A_4$

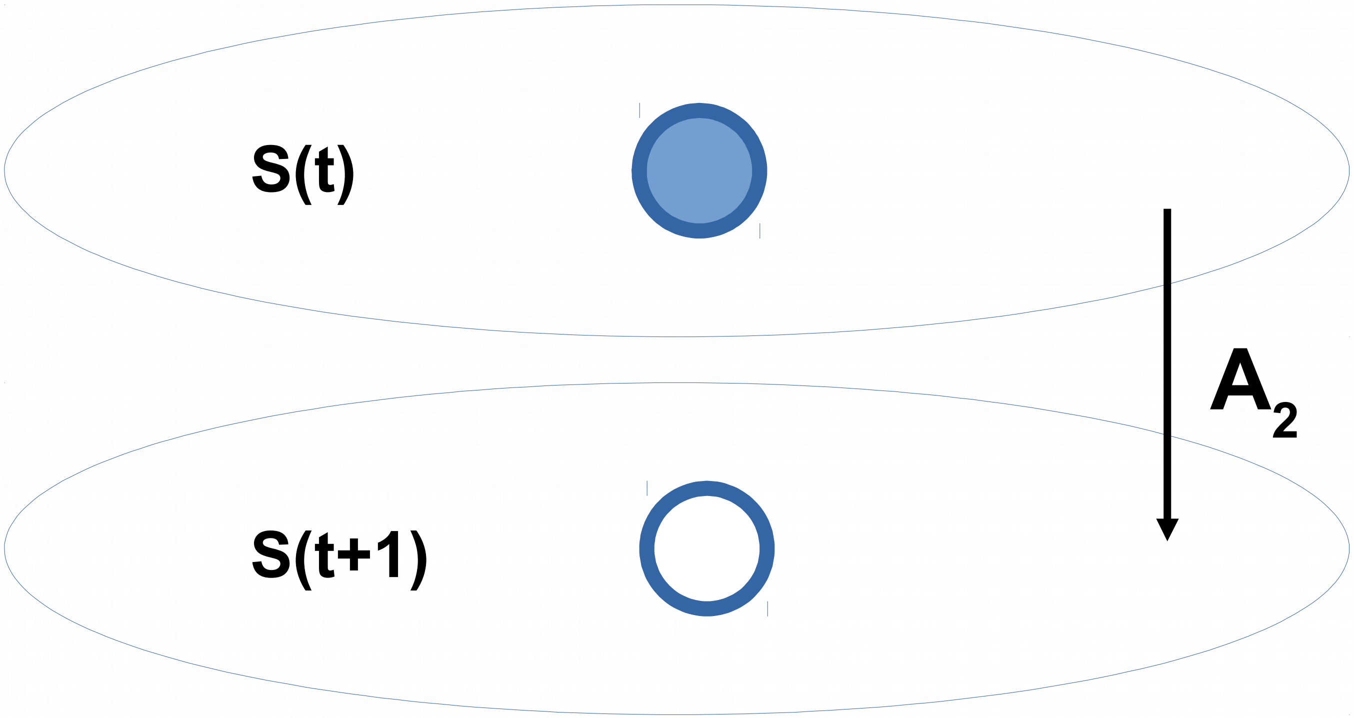
$$A_1 \equiv \exists!\, j : \neg a[i] \wedge m[j] \wedge b[j] = i$$
$$\wedge\, a'[i] \wedge \neg m'[j] \wedge b'[j] = i$$



34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

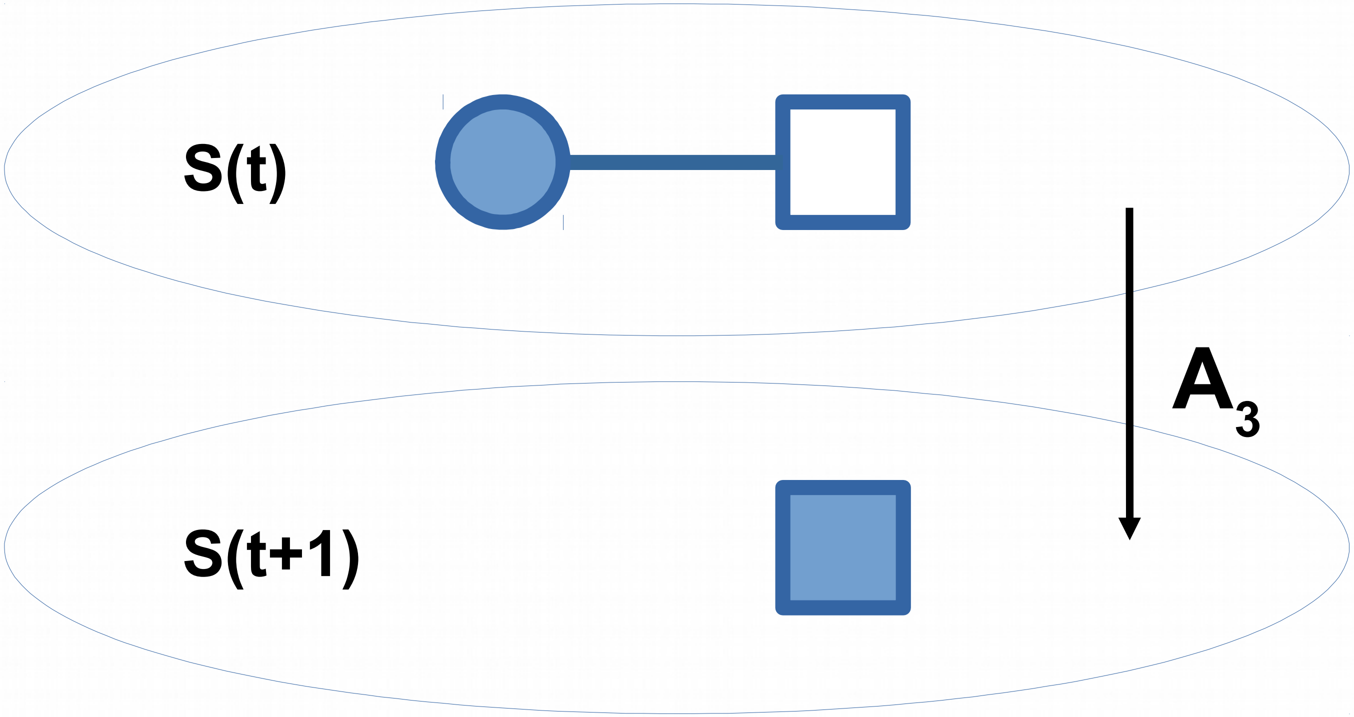
A2: ACTOR FINISHES THE EXECUTION

$$A_2 \equiv a[i] \wedge \neg a'[i]$$

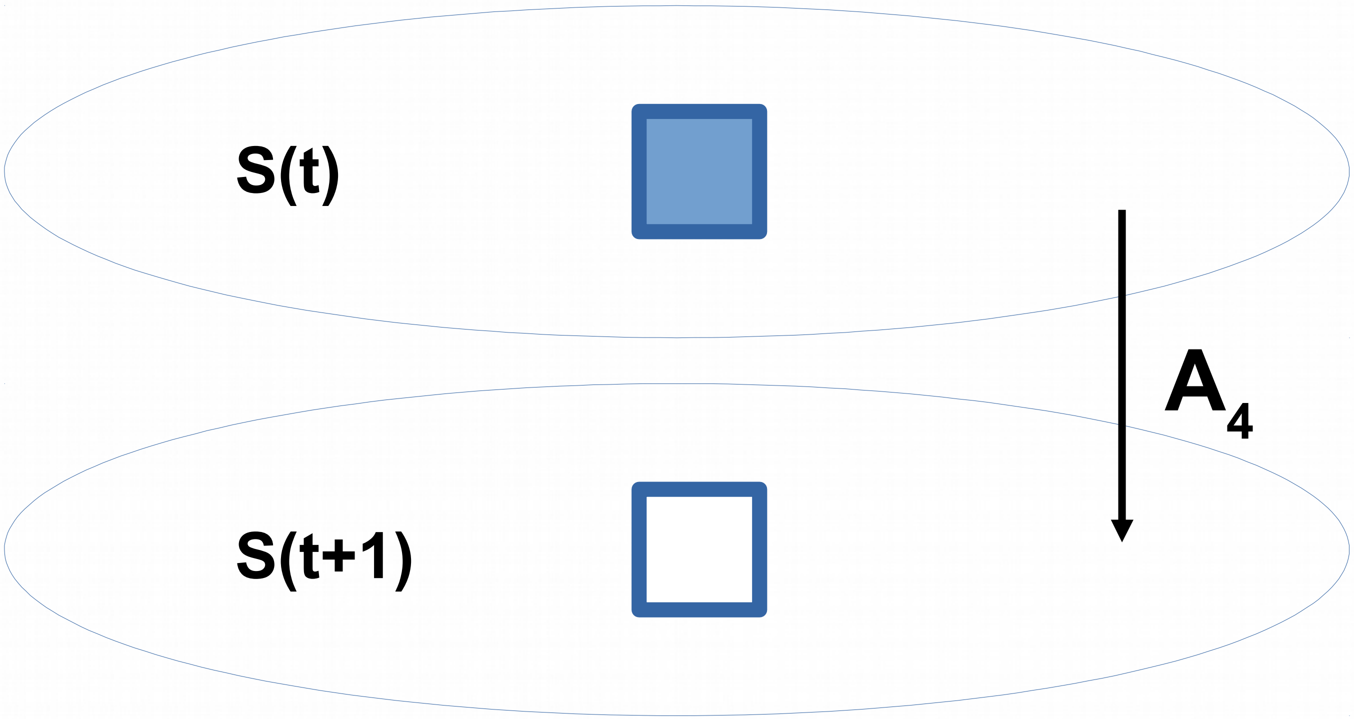


34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

$$A_3 \equiv \exists i : a[i] \wedge b[j] = i \wedge \neg m[j] \wedge m'[j]$$



$$A_4 \equiv m[j] \wedge \neg m'[j]$$



34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

$$I \equiv \exists i : a[i] \vee \exists j : m[j]$$

$$f_1 \equiv a[i] \qquad f_2 \equiv (m[j], b[j])$$

$\Box[A]_f$ means that the action $A \vee (f = f')$ executes for every pairs of system states

$WF_f(A)$ means that if the action ***A*** (that change variables ***f***) is enabled long enough it will finally be executed

$$
\begin{aligned}
S \equiv I &\wedge \Box [A_1 \vee A_2]_{f_1} \\
&\wedge \Box [A_3 \vee A_4]_{f_2} \\
&\wedge WF_{f_1}(A_2) \\
&\wedge WF_{f_2}(A_4)
\end{aligned}
$$

34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

The message handler procedure ***RECV(i,j)***
is called with the action

$$recv_{(call)}(i,j) \equiv \neg a[i] \wedge m[j] \wedge b[j] = i \\ \wedge a'[i] \wedge \neg m'[j] \wedge b'[j] = i$$

and returns with the action

$$recv_{(return)}(i) \equiv a[i] \wedge \neg a'[i]$$

of the actor run-time system ***S***

34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

Testing for the accessibility of a message

$$access(i, j) \equiv b[j] = i \wedge \neg m[j]$$

Sending a message to an actor

$$send(i, j) \equiv b'[j] = i \wedge m'[j]$$

(A) the handler ***RECV*(I,_)** can access actor state variables ***var***, if ***F*(*var*)=I**

(B) the handler ***RECV*(I,_)** can access message state variables ***var***, if ***F*(*var*)=J** and **access(I,J)**=true

(C) the handler ***RECV*(I,_)** can send a message with the call **send(_,J)**, if **access(I,J)**=true

34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

THE ACTOR RUN-TIME SYSTEM IMPLEMENTATION (1/3)

```
// engine
struct engine{ std::vector<message*> ready;};
// actor objects
struct actor{ void(*recv)(actor*,message*);};
// message objects
struct message{ actor*a; bool sending;};

inline void send(engine*e, actor*a, message*m){
	if (m->sending) return;
	m->sending = true;
	m->a = a;
	e->ready.push_back(m);
}
```

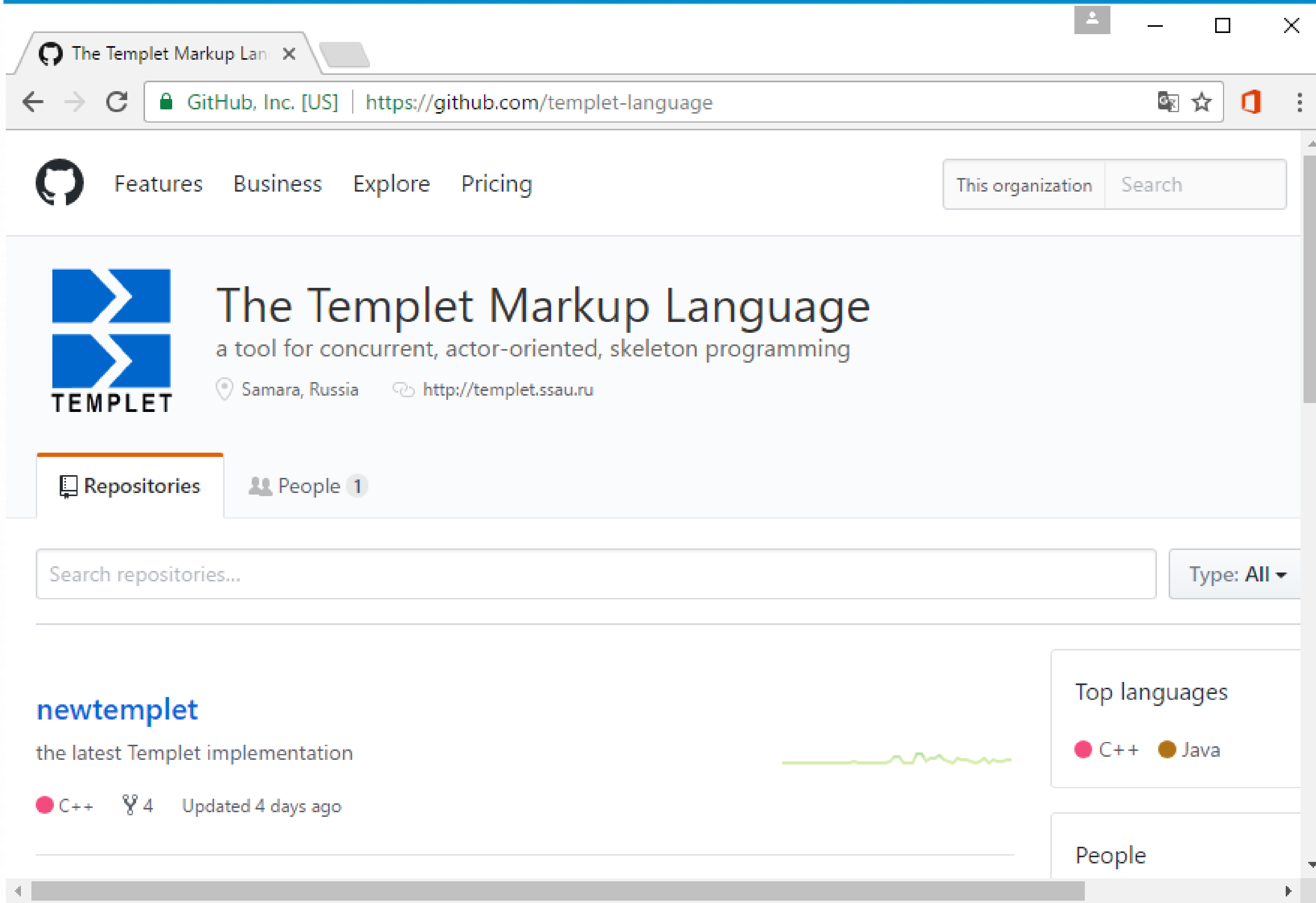
```
inline bool access(actor*a, message*m){
        return m->a == a && !m->sending;
}
inline void run(engine*e){
        size_t rsize;
        while (rsize = e->ready.size()){
                int n = rand() % rsize;
                auto it = e->ready.begin() + n;
                message*m = *it; e->ready.erase(it);
                m->sending = false;
                m->a->recv(m->a,m);
        }
}
```

34, Moskovskoye shosse, Samara, 443086, Russia, tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36 www.ssau.ru, e-mail: ssau@ssau.ru

For more details, please, see the poster section presentation:

S.Vostokin, E.Skoryupina
A Performance Analysis of Simple Runtime System for Actor Programming in C++

THE TEMPLET SYSTEM RUN-TIME AND APPLICATIONS



САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

THANK YOU

http://templet.ssau.ru

https://Github.com/templet-language

E-mail: Sergei Vostokin <easts@mail.ru>

34, Moskovskoye shosse, Samara, 443086, Russia
Tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36
www.ssau.ru, e-mail: ssau@ssau.ru