F.L. Lewis
Moncrief-O'Donnell Endowed Chair
Head, Controls & Sensors Group

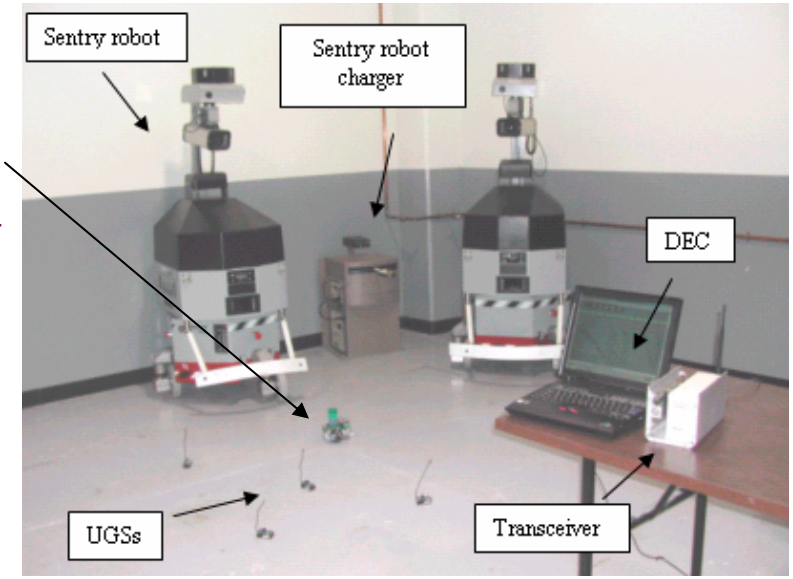
Automation & Robotics Research Institute
The University of Texas at Arlington

Wireless Sensor Networks

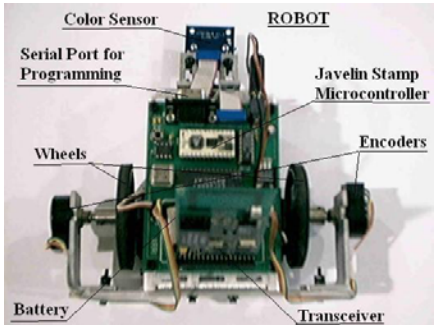
Talk available online at
http://ARRI.uta.edu/acs

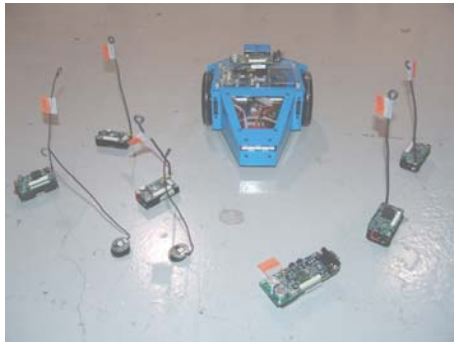
II. ARRI Distributed Intelligence & Autonomy Lab
DIAL

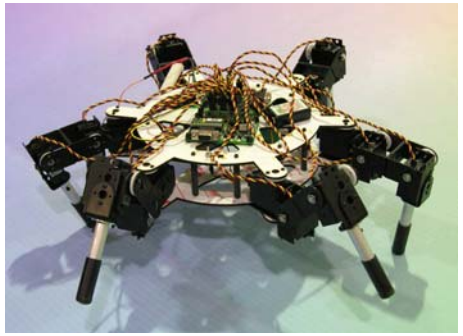
Small mobile Sensor- Dan Popa

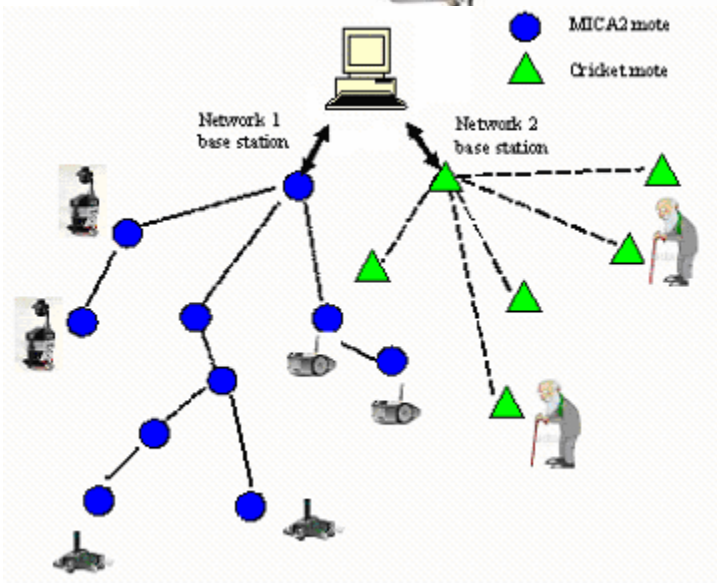


Unattended Ground Sensors

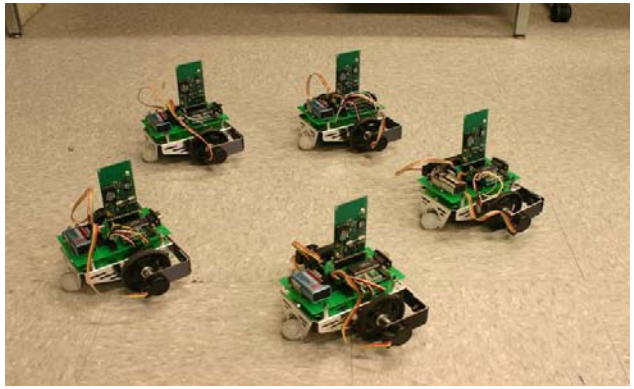








Testbed containing MICA2 network (circle), Cricket network (triangle), Sentry robots, Garcia Robots & ARRI-bots (Dan Popa)



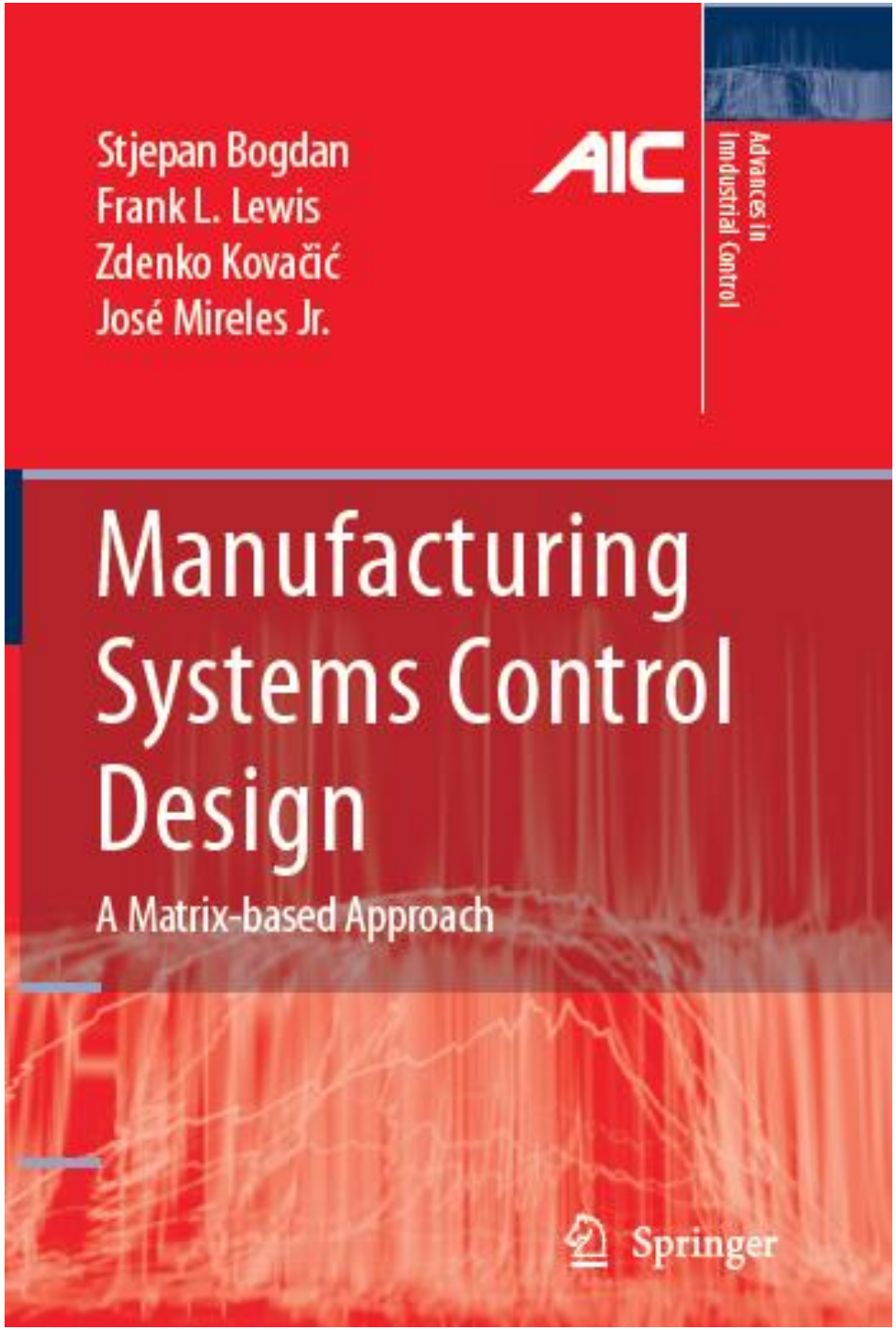
Deadlock free dynamic resource assignment in multi-robot systems with multiple missions: a matrix-based approach

by

Prasanna Ballal, Vincenzo Giordano, Frank Lewis

Automation & Robotics Research Institute,
University of Texas at Arlington

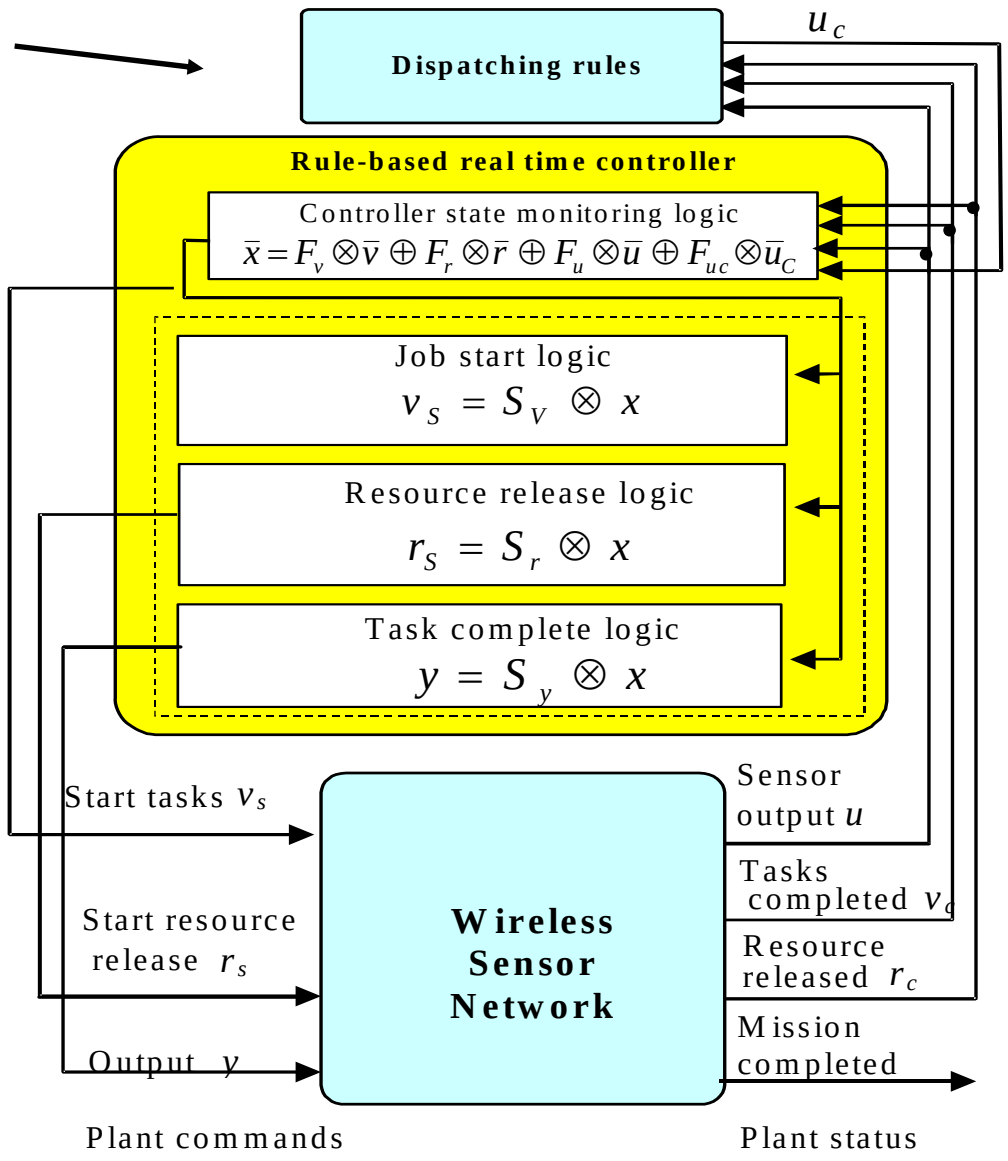
Distributed Intelligence and Autonomy Laboratory (DIAL)


Stjepan Bogdan
Frank L. Lewis
Zdenko Kovačić
José Mireles Jr.
AIC
Advances in Industrial Control
Manufacturing Systems Control Design
A Matrix-based Approach
Springer

U.S. Patent

Discrete event controller

User interface:definition of mission planning, resource allocation, priority rules



Decision-making

commands

Sensor readings

events

Matrix Formulation: Definition

multiply = AND & addition = OR
overbar = negation

Compare with $x_{k+1}=Ax_k+Bu_k$

Logical state equation

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

State vector

Task vector
1= job complete

Resource vector
1= resource available

Input vector
1= event detected

Control input
priority sequencing
deadlock avoidance
etc

Output equations

Compare with $y_k=Cx_k$

Job Start Equation: $v_s = S_v x$

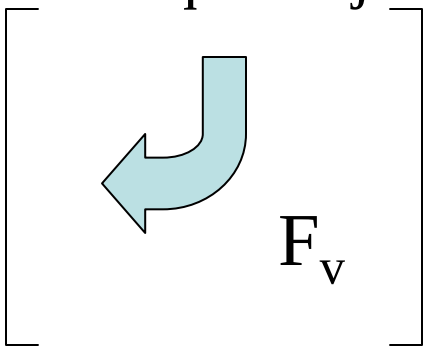
Resource Release Equation: $r_s = S_r x$

Product Output Equation: $y = S_y x$

Meaning of Matrices

Prerequisite jobs

Next job

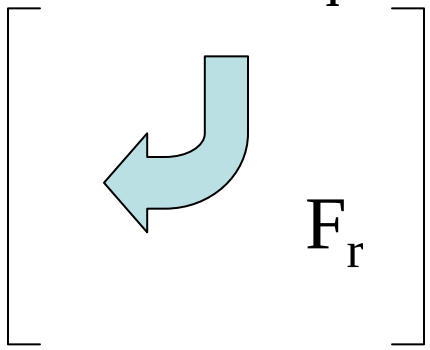


Task Sequencing Matrix

Steward

Resources required

Next job

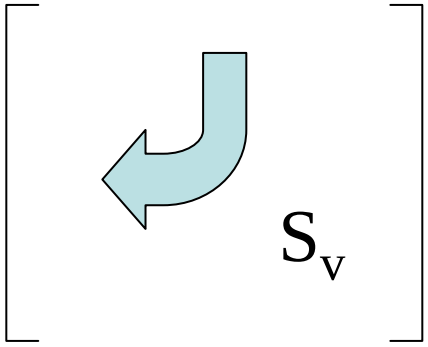


Resource Requirements Matrix

Kusiak

Conditions fulfilled

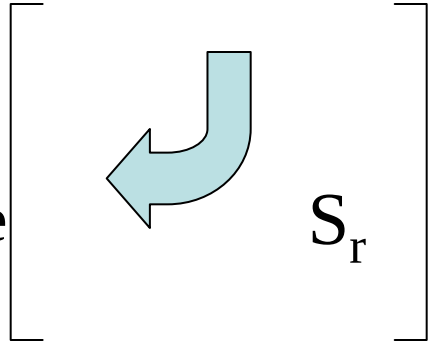
Next job



Task Starting Matrix

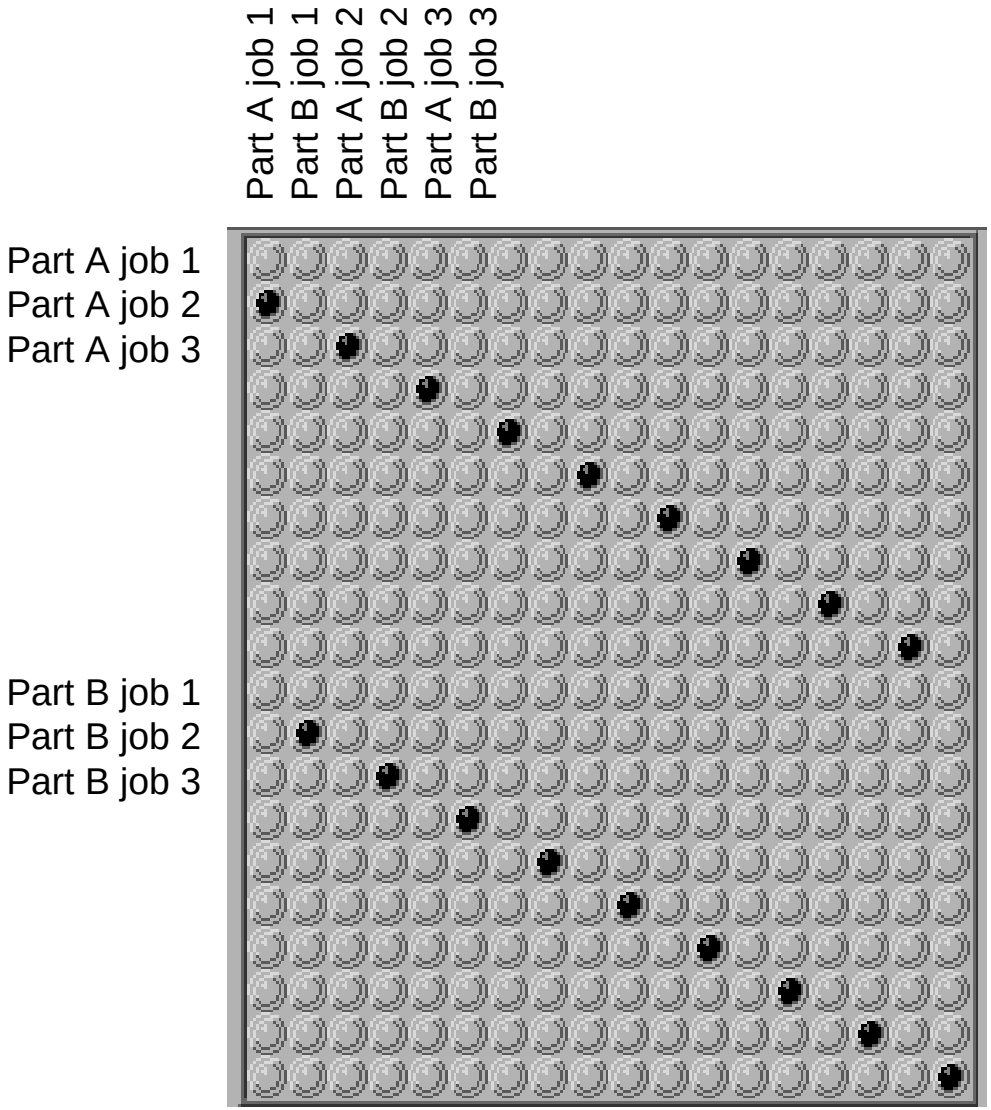
Conditions fulfilled

Release resource

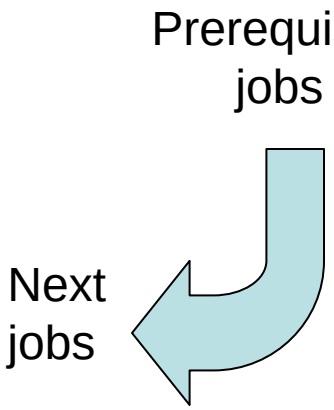


Resource Releasing Matrix

Construct Job Sequencing Matrix F_v

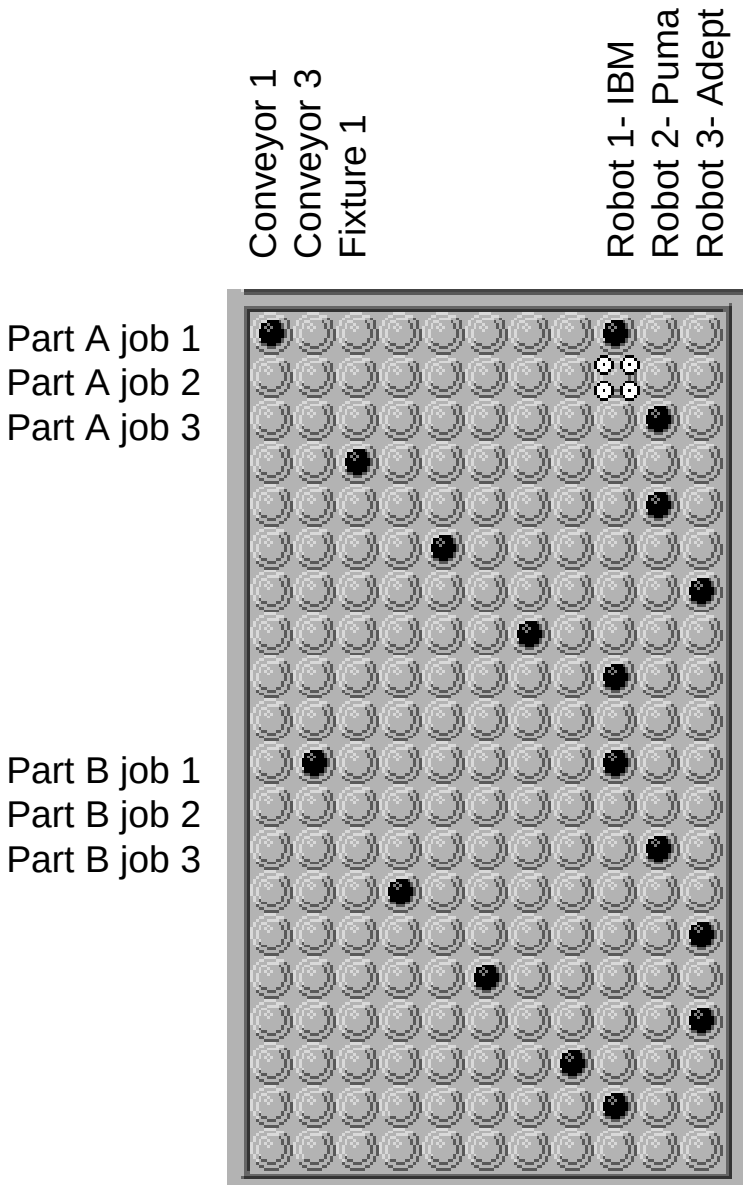


Used by Steward in Manufacturing Task Sequencing



Contains same information as the Bill of Materials (BOM)

Construct Resource Requirements Matrix F_r



Used by Kusiak in Manufacturing Resource Assignment

Contains information about factory resources

Prerequisite resources

Next jobs

OR / AND Matrix Algebra

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c$$

Example

$$\bar{x} = [1 \quad 0 \quad 1] \begin{bmatrix} \bar{a} \\ \bar{b} \\ \bar{c} \end{bmatrix} = (1 \wedge \bar{a}) \vee (0 \wedge \bar{b}) \vee (1 \wedge \bar{c})$$

$$x = \overline{(1 \wedge \bar{a}) \vee (0 \wedge \bar{b}) \vee (1 \wedge \bar{c})}$$

$$x = \overline{(1 \wedge \bar{a})} \wedge \overline{(0 \wedge \bar{b})} \wedge \overline{(1 \wedge \bar{c})}$$

$$x = (0 \vee a) \wedge (1 \vee b) \wedge (0 \vee c)$$

$$x = a \wedge c$$

Easy to implement OR/ AND algebra in MATLAB

Matrix multiply

C= AB

```
for i= 1,I
   for j= 1,J
      c(i,j)=0
      for k= 1,K
          c(i,j)= c(i,j) .OR. ( a(i,k) .AND. b(k,j) )
      end
    end
end
```

Relation to Max-Plus Algebra

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

State equation

$$V_s = S_v x$$
$$r_s = S_r x$$
$$y = S_y x$$

Output equations

OPERATIONS IN OR-AND ALGEBRA

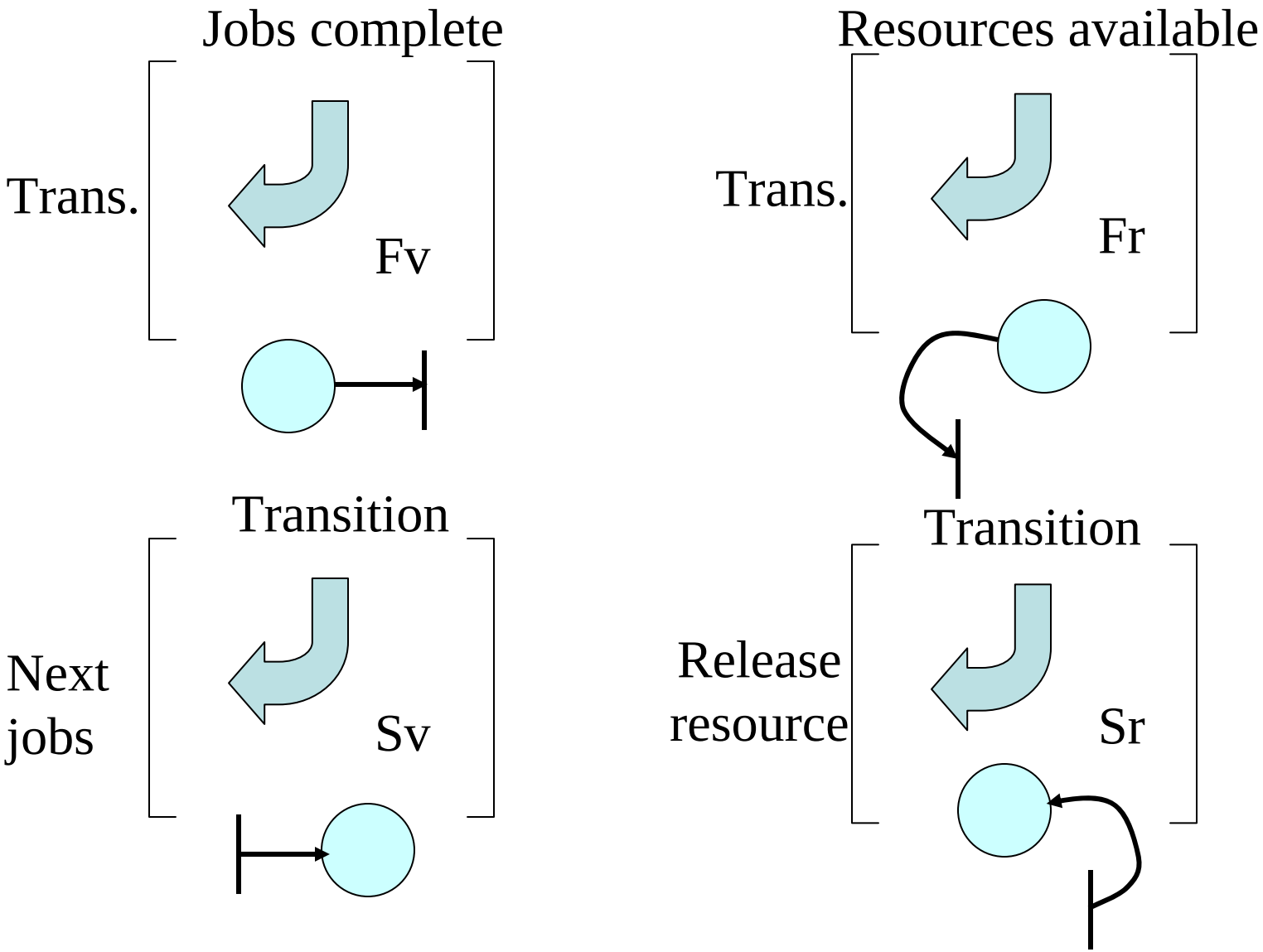
Define diagonal timing matrices. Then max plus is

$$x' = S_v T_v F_v x + S_r T_r F_r r$$

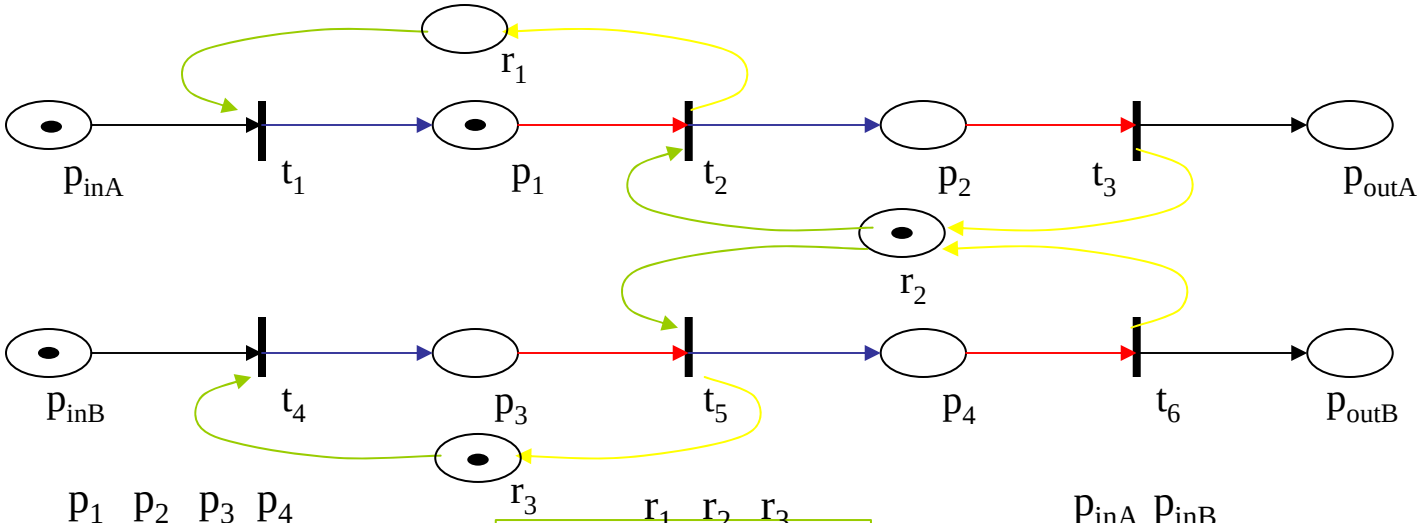
OPS. IN MAX-PLUS ALGEBRA

Can also include nonlinear terms- correspond to decisions

Relation to Petri Nets



UTA



$$F_v = \begin{array}{c} \begin{matrix} p_1 & p_2 & p_3 & p_4 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$F_r = \begin{array}{c} \begin{matrix} r_1 & r_2 & r_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$F_u = \begin{array}{c} \begin{matrix} p_{inA} & p_{inB} \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{array}$$

$$S_v^T = \begin{array}{c} \begin{matrix} p_1 & p_2 & p_3 & p_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$S_r^T = \begin{array}{c} \begin{matrix} r_1 & r_2 & r_3 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \end{array}$$

$$S_y^T = \begin{array}{c} \begin{matrix} p_{outA} & p_{outB} \end{matrix} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \end{array}$$

ARRI

Complete DE Dynamical Description

DE state equation

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

Activity Completion Matrix

$$F = [F_u \quad F_v \quad F_r \quad F_y]$$

Activity Start Matrix

$$S = [S_u{}^T \quad S_v{}^T \quad S_r{}^T \quad S_y{}^T]$$

PN incidence matrix

$$M = S^T - F = [S_u{}^T - F_u,\ S_v{}^T - F_v,\ S_r{}^T - F_r,\ S_y{}^T - F_y]$$

Marking transition equation

$$m(t+1) = m(t) + M^T x = m(t) + [S^T - F]^T x$$

$$p = \left\lfloor u^T \quad v^T \quad r^T \quad y^T \right\rfloor^T$$

Include Process Times in Places

Split up marking vector

$$m(t) = m_a(t) + m_p(t)$$

Add tokens

$$m_p(t+1) = m_p(t) + S^T\, x(t)$$

Wait for process durations

$$T = [O,\ vtimes^T, rtimes^T, O]^T$$

$$T_{pend}(t+1) = diag\{m_p(t)\}[T_{pend}(t) - t_{sample}] + diag\{S^T x(t)\}T$$

Take away tokens

$$m_a(t+1) = m_a(t) - F\, x(t)$$

Allows easy MATLAB simulation of DE systems

1. Rules- fire transitions

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

Controller

2. Add tokens to places

$$m_p(t+1) = m_p(t) + S^T\, x(t)$$

3. Wait until jobs finish

Duration time counting routine

4. Take tokens from places

$$m_a(t+1) = m_a(t) - F\, x(t)$$

5. Find updated vectors for DE state equation

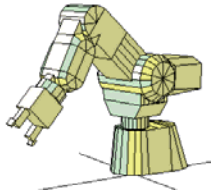
$$m(t+1) = m_a(t+1) + m_p(t+1)$$

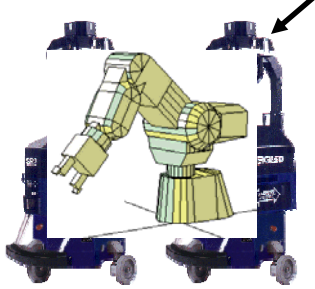
6. $\left\lfloor u^T \quad v^T \quad r^T \quad y^T \right\rfloor^T = \overline{\overline{m(p)}}$

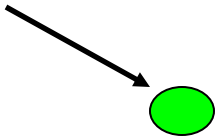
System model

DEC for WSN

Programmable Missions







Fast Programming of Missions

Mission1-
Task sequence

mission1	notation	Description
Input 1	u^1	*UGS1* launches chemical alert
Task 1	$S4m^1$	*UGS4* takes measurement
Task 2	$S5m^1$	*UGS5* takes measurement
Task 3	$R1gS2^1$	*R1* goes to *UGS2*
Task 4	$R2gA^1$	*R2* goes to location A
Task 5	$R1rS2^1$	*R1* retrieves *UGS2*
Task 6	$R1lis^1$	*R1* listens for interrupts
Task 7	$R1gS1^1$	*R1* gores to *UGS1*
Task 8	$R2m^1$	*R2* takes measurement
Task 9	$R1dS2^1$	*R1* deploys *UGS2*
Task 10	$R1m^1$	*R1* takes measurement
Task 11	$S2m^1$	*S2* takes measurement
output	y^1	Mission 1 completed

Mission 2-
Task sequence

Mission2	notation	Description
input	u^2	*UGS3* batteries are low
Task 1	$S1m^2$	*UGS1* takes measurement
Task 2	$R1g\ S3^2$	*R1* goes to *UGS3*
Task 3	$R1cS3^2$	*R1* charges *UGS3*
Task 4	$S3m^2$	*UGS3* takes measurement
Task 5	$R1dC^2$	*R1* docks the charger
output	y^2	Mission 2 completed

Mission 1 matrices

$$F_v^1 = \begin{array}{c} \\ x_1^1 \\ x_2^1 \\ x_3^1 \\ x_4^1 \\ x_5^1 \\ x_6^1 \\ x_7^1 \\ x_8^1 \\ x_9^1 \end{array} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

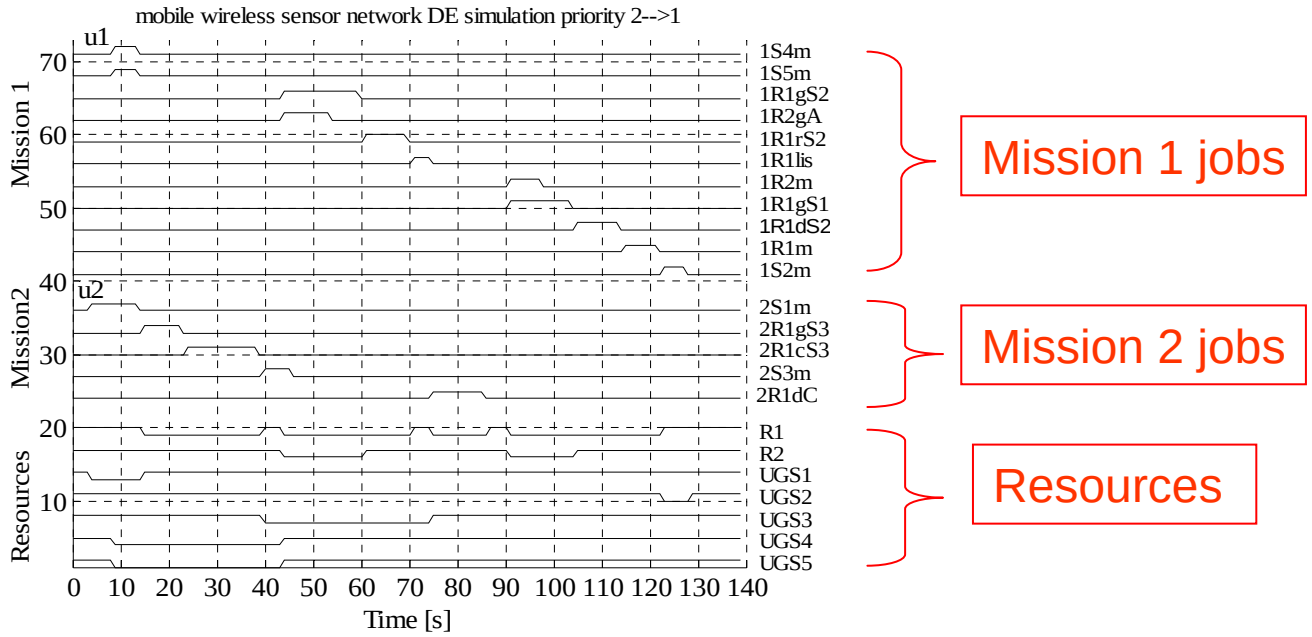
$$F_r^1 = \begin{array}{c} \\ x_1^1 \\ x_2^1 \\ x_3^1 \\ x_4^1 \\ x_5^1 \\ x_6^1 \\ x_7^1 \\ x_8^1 \\ x_9^1 \end{array} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Mission 2 matrices

$$F_v^2 = \begin{array}{c} \\ x_1^2 \\ x_2^2 \\ x_3^2 \\ x_4^2 \\ x_5^2 \\ x_6^2 \end{array} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$F_r^2 = \begin{array}{c} \\ x_1^2 \\ x_2^2 \\ x_3^2 \\ x_4^2 \\ x_5^2 \\ x_6^2 \end{array} \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

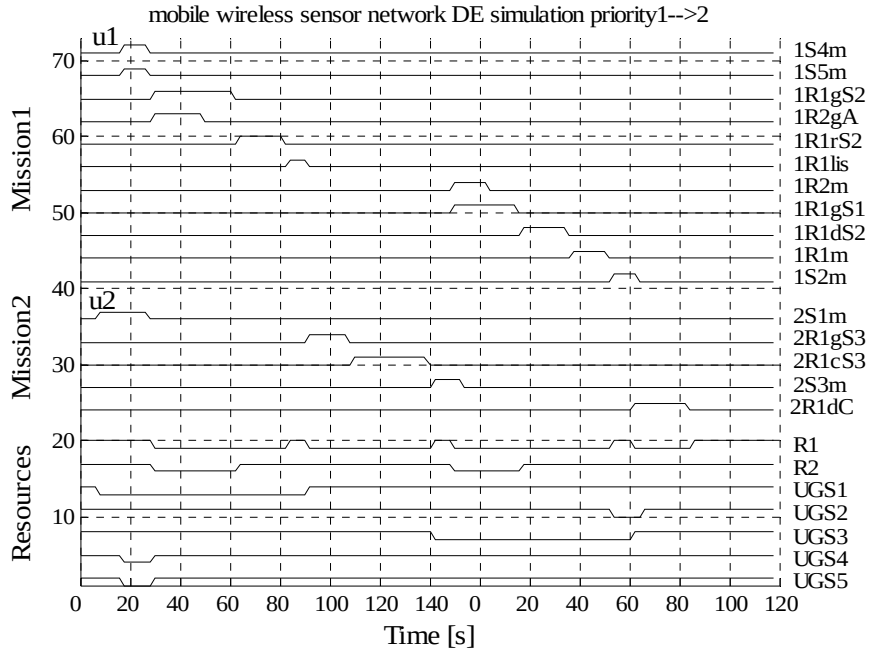
Simulation Results



Event traces

- Up means task in progress
- Down ☹ means resource in use

Simulation 2 –
change mission priority



DEC Implementation

DE controller on PC

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

System model

$$m_p(t+1) = m_p(t) + S^T\ x(t)$$

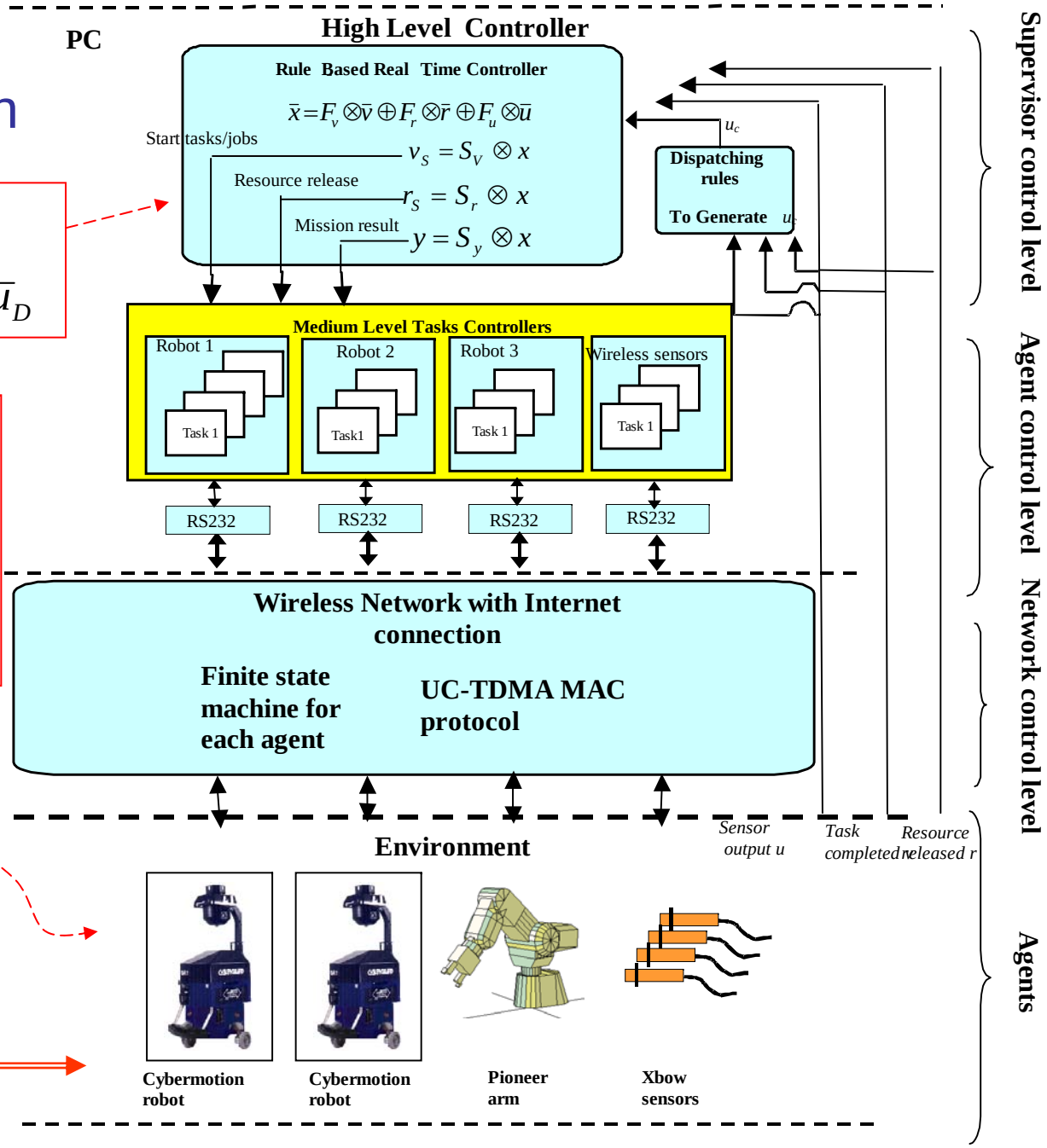
Duration times

$$m_a(t+1) = m_a(t) - F\ x(t)$$

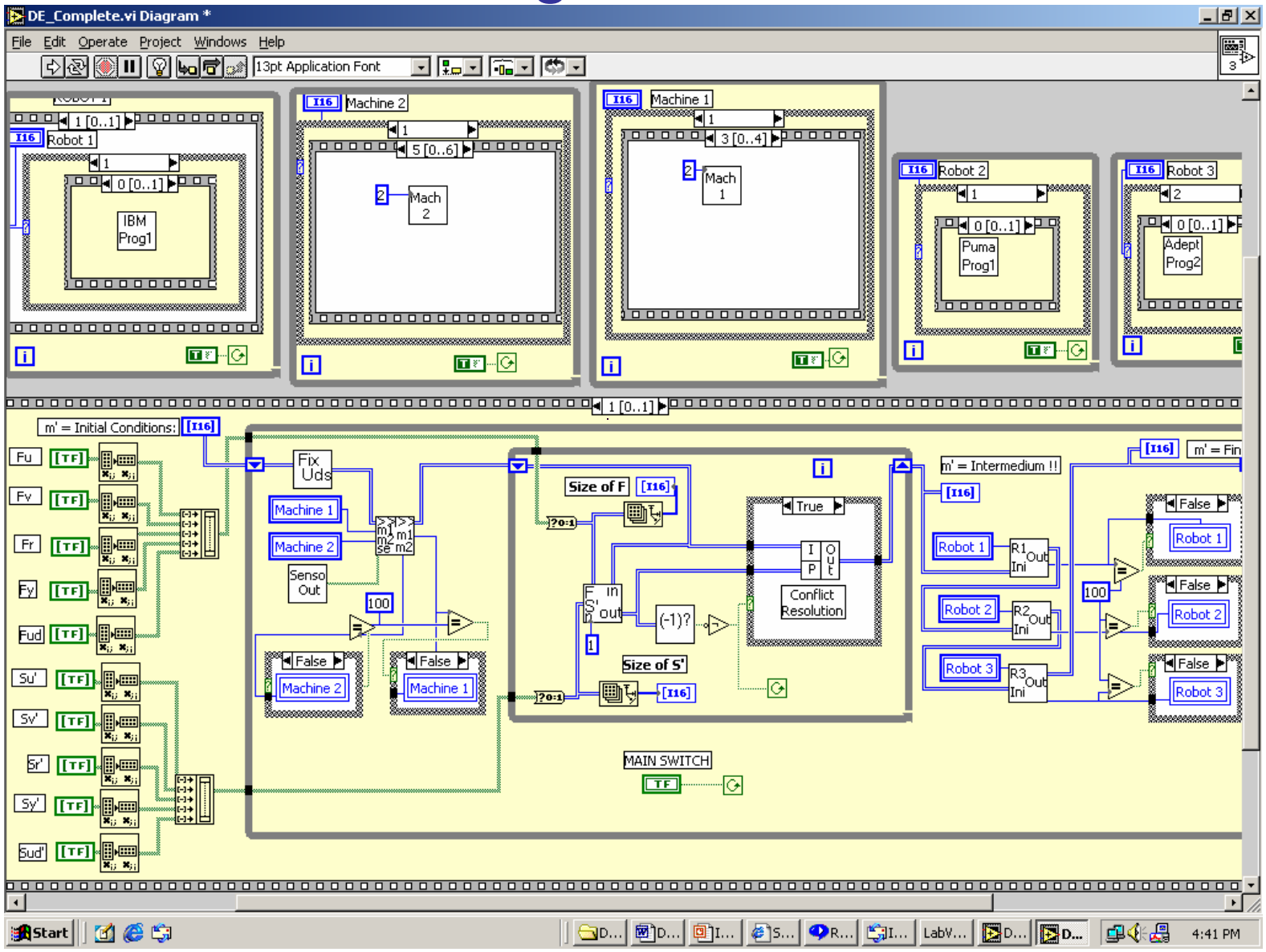
$$m(t+1) = m_a(t+1) + m_p(t+1)$$

Replaced by actual WSN

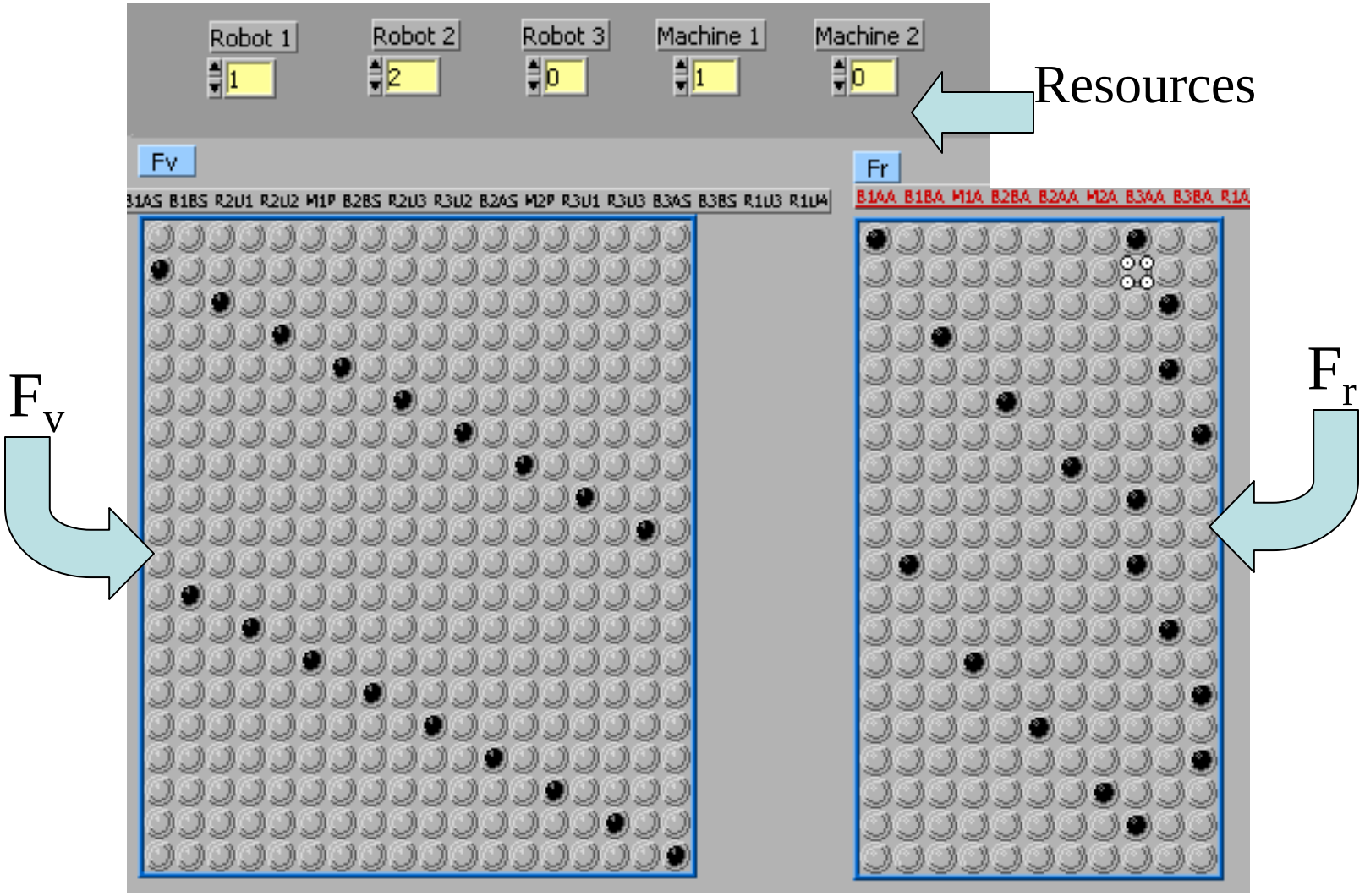
events



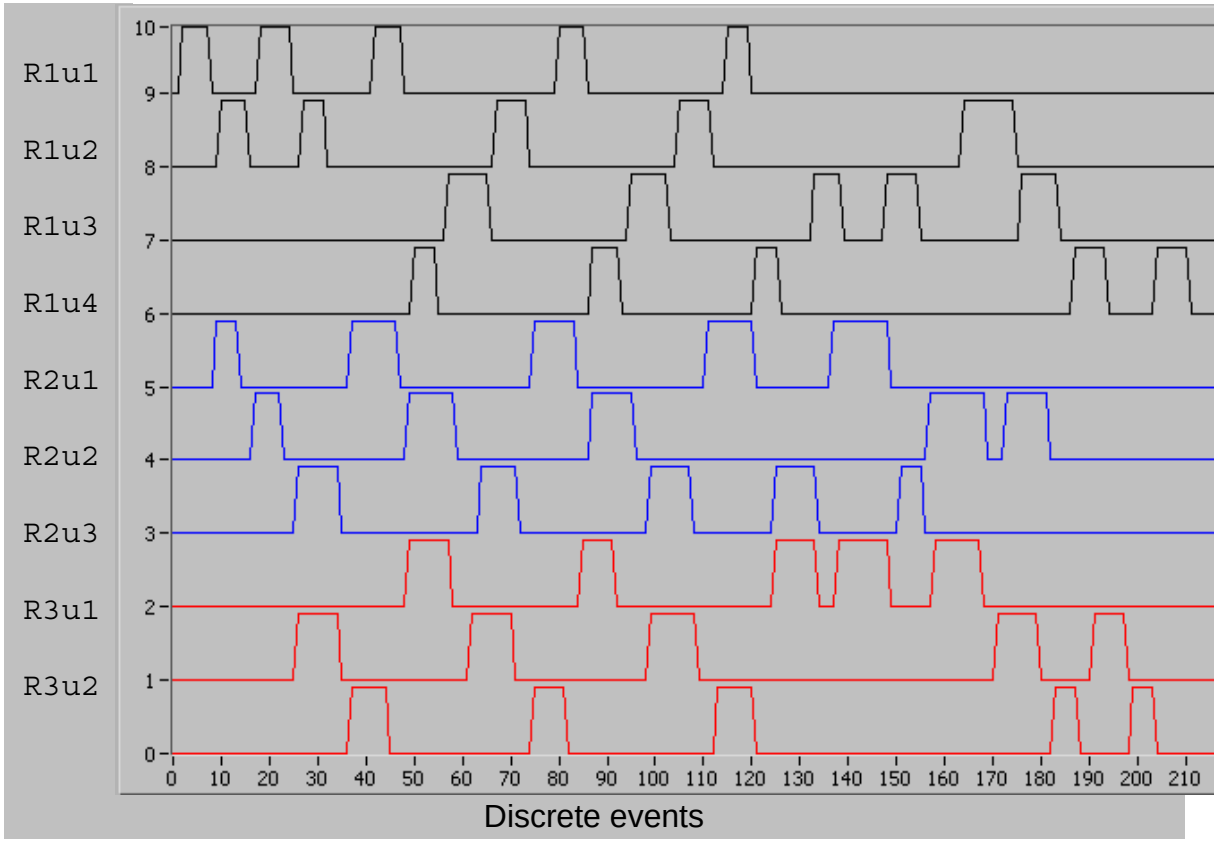
LabVIEW diagram of Controller



LabVIEW Controller's interface:



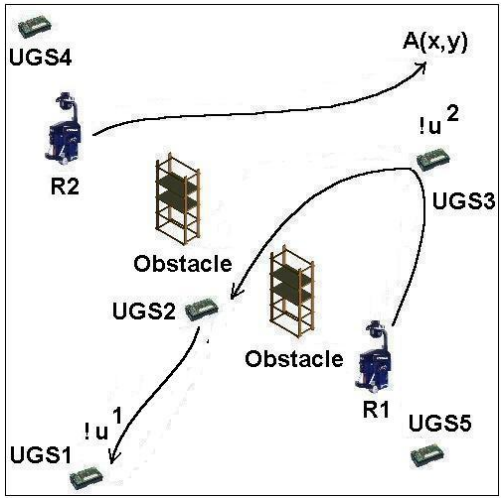
Results of LabVIEW Implementation on Actual Workcell

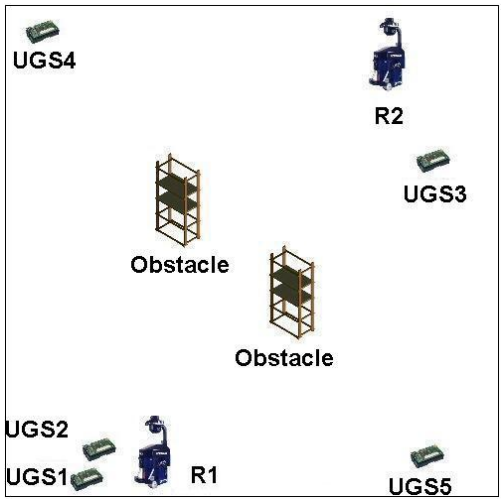


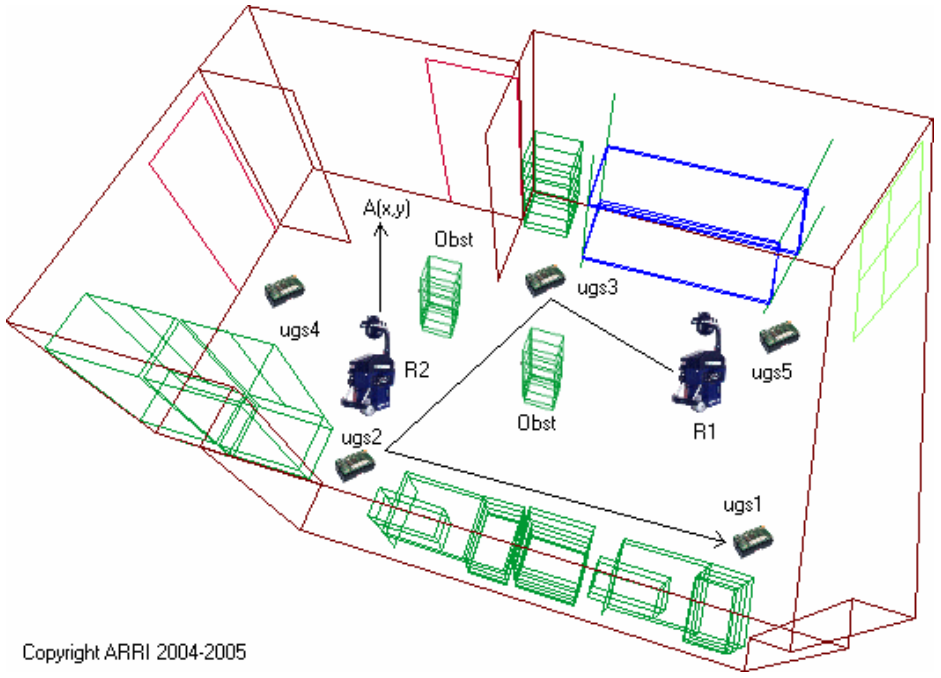
Compare with MATLAB simulation!

We can now simulate a DE controller and then implement it,
Exactly as for continuous state controllers!!

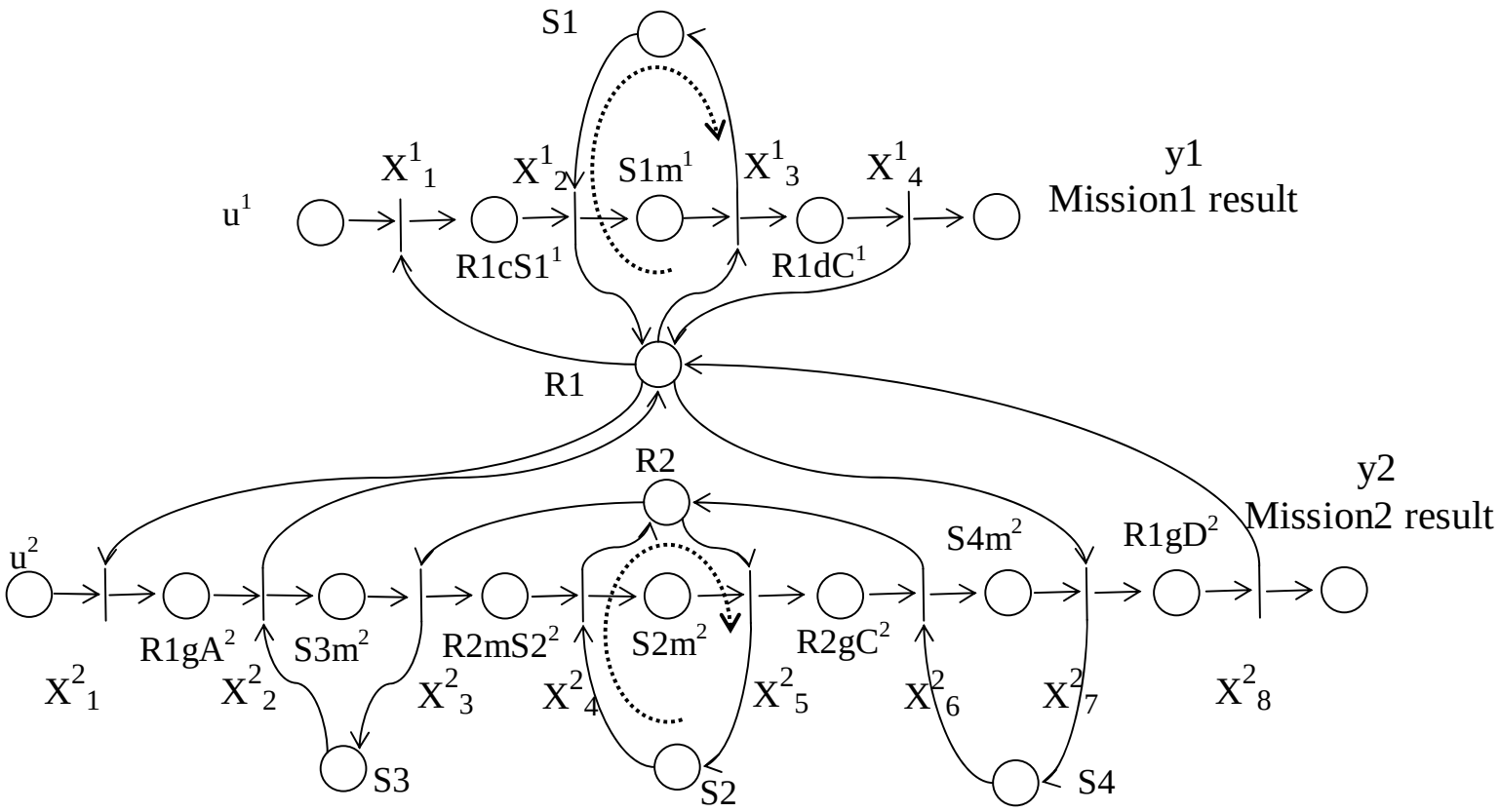
Schematic Event Sequence for Mission Performance



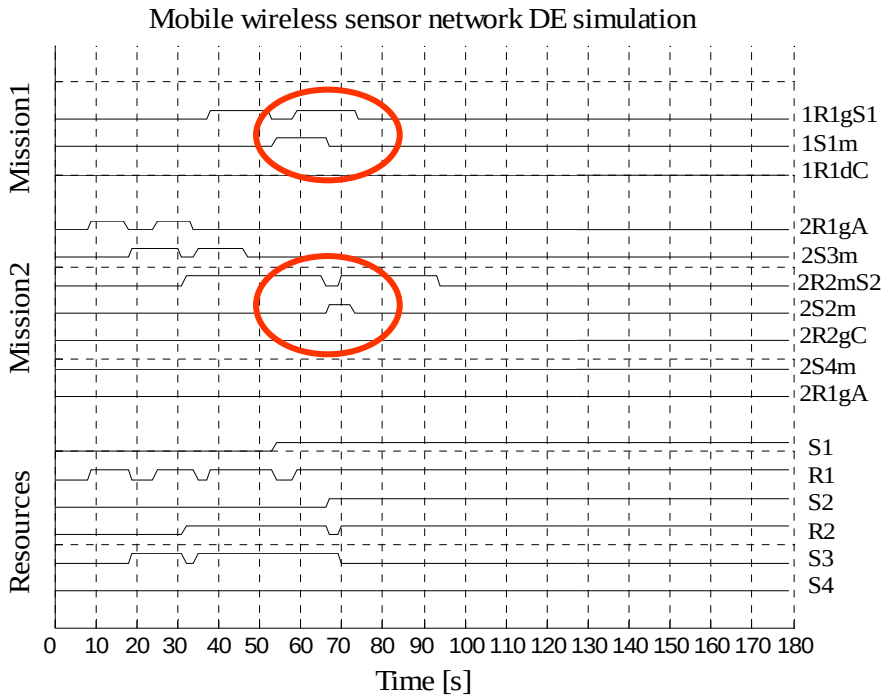




Shared resources→circular waits



Simulation results



Deadlock!

Event time trace

One Step Look Ahead Deadlock Avoidance Policy

Some Definitions:

- **Circular Waits**
  -Circular waits (CW) among resources are a set of resources *ra, rb,… rw* whose wait relationship among them are *ra*→*rb*→…→*rw* and *rw*→*ra*.

- **Simple CW**
  -Simple CWs (sCW) are primitive CWs which do not contain other CWs.

- **Critical siphons, S_c**
  - The critical siphon of a CW is the smallest siphon containing the CW. Note that if the critical siphon ever becomes empty, the CW can never again receive any tokens. This is, the CW has become a circular blocking.

One Step Look Ahead Deadlock Avoidance Policy ctd…

- **Siphon job set J_s**

  -Siphon-job set, J_s, is the set of jobs which, when added to the set of resources contained in CW *C*, yields the critical siphon.

- **Critical Subsystems J_o**

  -The critical subsystems of the CW *C*, are the ***job sets*** $J(C)$ from that *C* not contained in the siphon-job set J_s of C. That is $J_o = J(C) \setminus J_s$.

Deadlock Analysis

The *Circular Waits* are key to deadlock analysis (Wysk).
CW are difficult to find from the Petri Net.

The CW are given directly from our matrices by the Wait Relation Matrix

$$G_W = S_r F_r$$ (in AND/OR algebra)

i.e. if *gij*=1 then resource *j* waits for resource *i*

Use string algebra to find a matrix ***C***, wherein each column represents a circular wait.

When a CW becomes blocked, one has Deadlock. To avoid this, examine the *Critical Siphons*. CS are difficult to find using Petri Nets.

The CS are given directly from our matrices by the columns of matrix

$$Sc = \begin{bmatrix} C_s^T S_r F_v \wedge \overline{C^T F_r^T F_v} \\ C \end{bmatrix}$$

Where vector C_s is the projection of *C* into the shared resources in the CW

and $\wedge$ denotes the element-by-element matrix 'and' operation.

One Step Look Ahead Deadlock Avoidance Policy ctd…

Critical Subsystems J_o

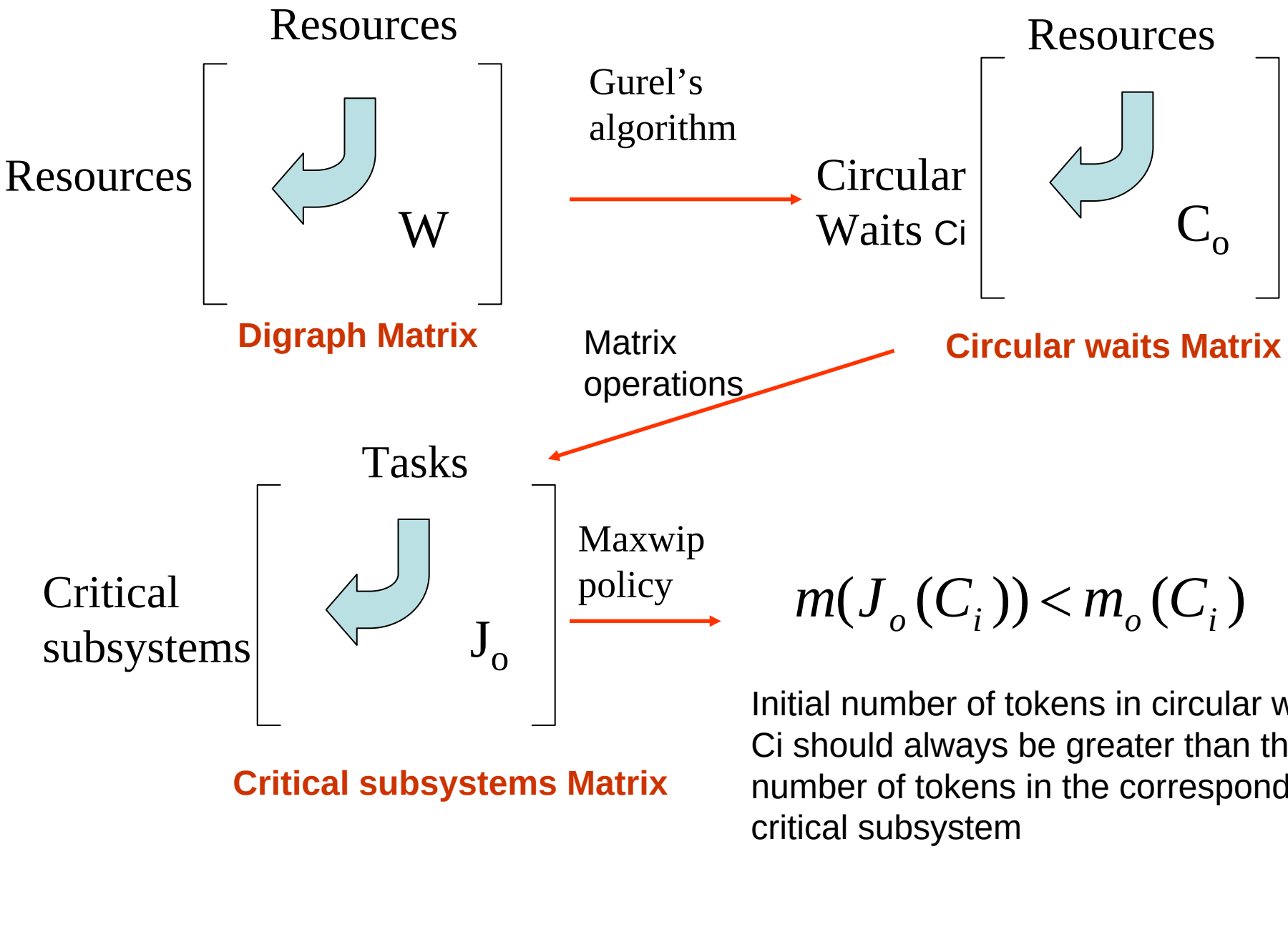
Matrix formulation for J_o

$$J_o = ({}_d CF_v) \wedge (C_d F_v) = (C_d S_v^T) \wedge (C_d F_v)$$

Deadlock Avoidance Strategy- MAXWIP Policy

$$m(J_o(C_i)) < m_o(C_i)$$

Deadlock analysis



Deadlock avoidance

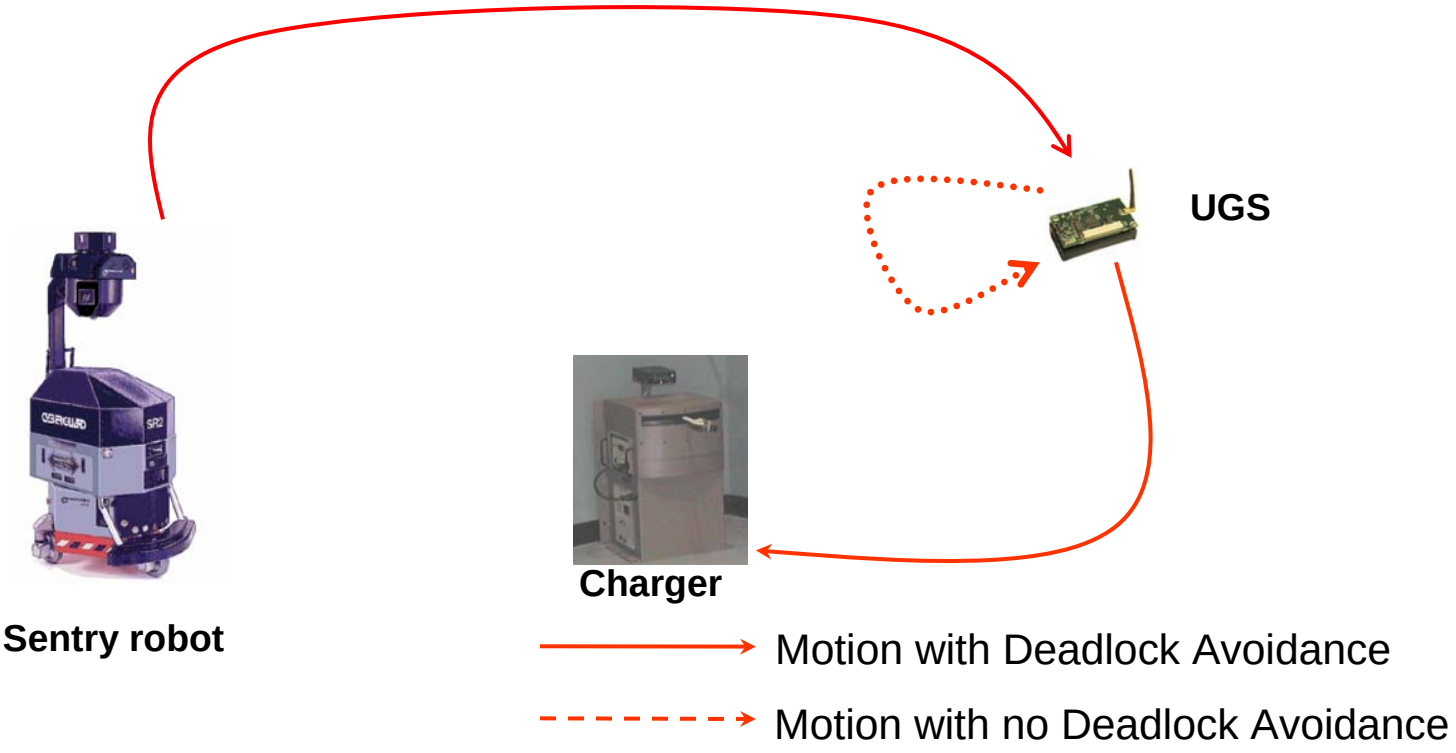
$$C_o = \begin{array}{c} \begin{matrix} S1 & S2 & S3 & S4 & R1 & R2 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Circular wait matrix

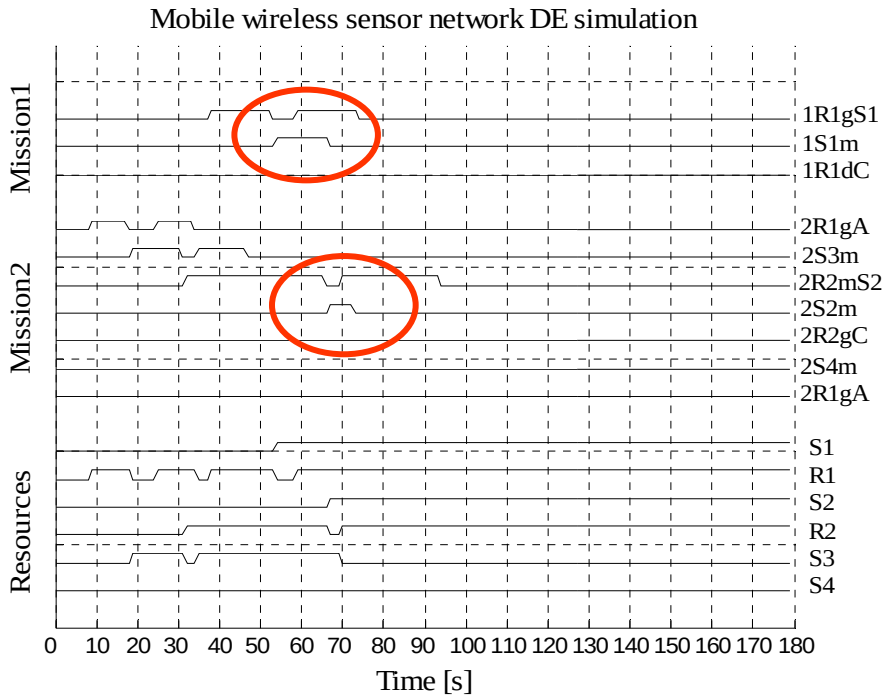
$$J_o = \begin{array}{c} \begin{matrix} R1cS1 & R1dC & S3m & S2m & S4m \\ S1m & R1gA & R2mS2 & R2gC & R1gD \end{matrix} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Critical subsystem matrix

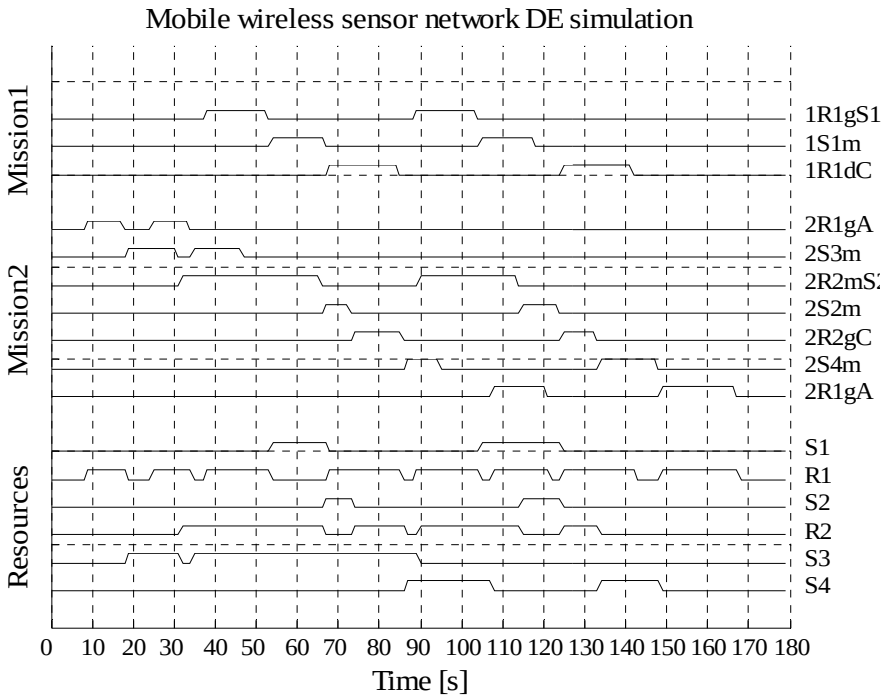
Deadlock avoidance



Simulation results



Event time trace without deadlock avoidance



Event time trace with deadlock avoidance

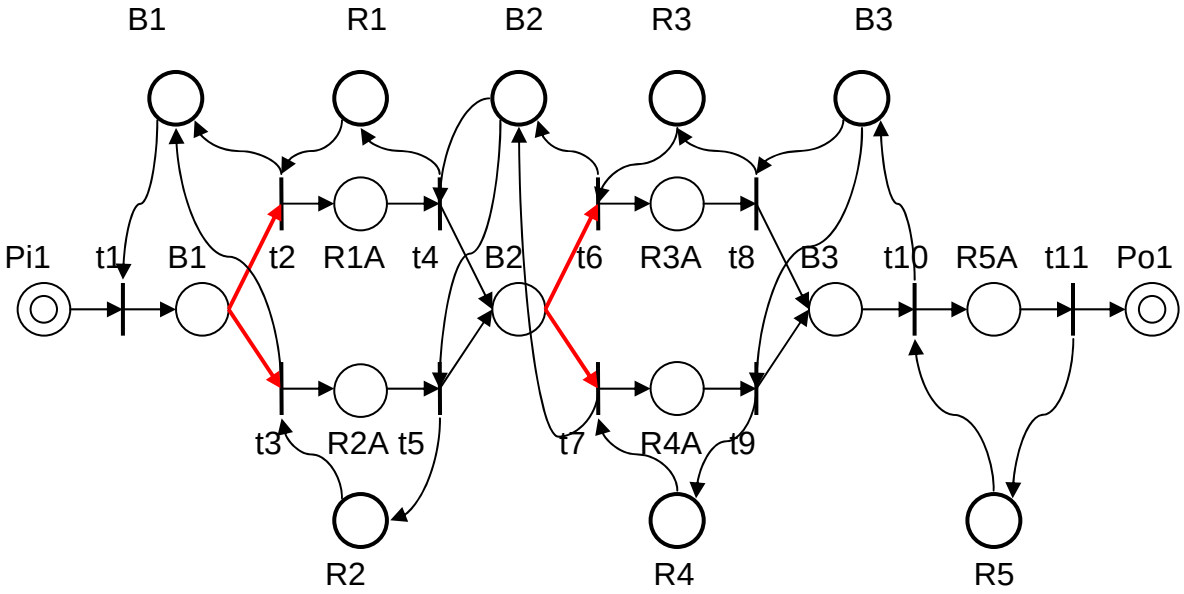
New Work—
Modify resource assignments on-line as sensor nodes fail or are added

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

Add or modify columns of F_r

Deadlock avoidance still works if there are no Key resources-
1-step look ahead.

New Work --
Deadlock Avoidance for Free-Choice Routing Systems



Choices in routing- select resource for next job

Problems with the old formulae-
need to make critical subsystems include alternate routing resources

New Work-

A new matrix algebra to implement Dempster-Shafer decisions

Make the state equation

$$\bar{x} = F_v \bar{v}_c + F_r \bar{r}_c + F_u \bar{u} + F_D \bar{u}_D$$

Implement the DS Rule of Combination

$$m_{1,2}(A) = \frac{\sum_{B_i \cap C_j = A} m_1(B_i) m_2(C_j)}{\sum_{B \cap C \neq \phi} m_1(B_i) m_2(C_j).}$$

Make the output equations

$$v_s = S_v x \qquad r_s = S_r x$$

Implement

Belief

$$Bel(A) = \sum_{B \subset A} m(B)$$

Plausibility

$$Pl(a) = \sum_{B \cap A \neq 0} m(B)$$