A CHANNEL PREDICTIVE PROPORTIONAL FAIR SCHEDULING ALGORITHM

Hans Jørgen Bang,
Torbjörn Ekman, David Gesbert
Tunisia May 2005

Overview

- Introduction
- Proportional Fair Scheduling
- Predictive Scheduling
  - Iterative algorithm
- Simulations
  - Fairness measure
- Conclusion

Introduction

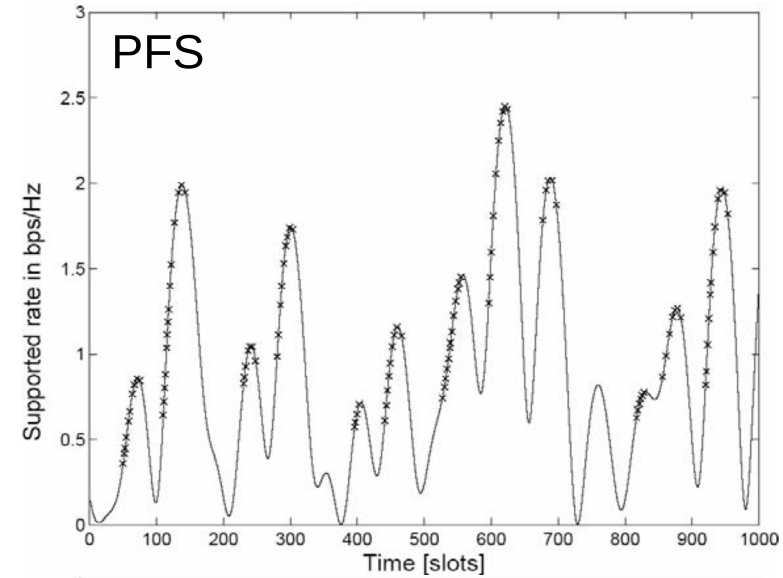
- Multi-user diversity scheduling
  - The supported rates for each user vary
  - Schedule to increase system throughput
- Channel prediction
  - Future supported rates can be estimated
- Improved throughput-fairness trade-off

Throughput-fairness trade-off

- Fundamental trade-off between total cell throughput and fairness
- Max SNR scheduling
  - Max throughput
  - Relies only on the current channel state
  - Fair over infinite time horizon for equal channel statistics (otherwise normalized max SNR scheduling)
- Tighter fairness constraints
  - Leads to reduced throughput
  - Gains can be obtained by using fading predictions

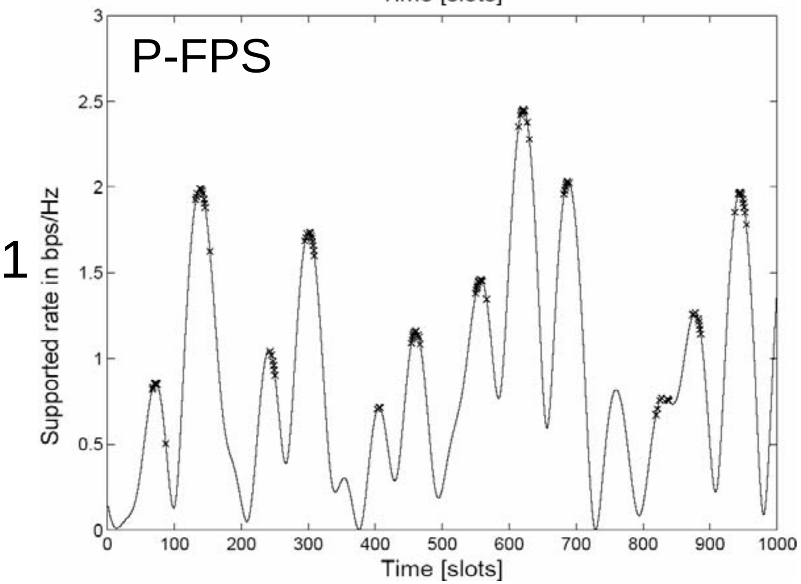
A Qualitative Comparison

- Proportional Fair Scheduling (PFS) v.s. Predictive PFS
- Scheduling around the peaks instead for on the flanks.
- Improved throughput

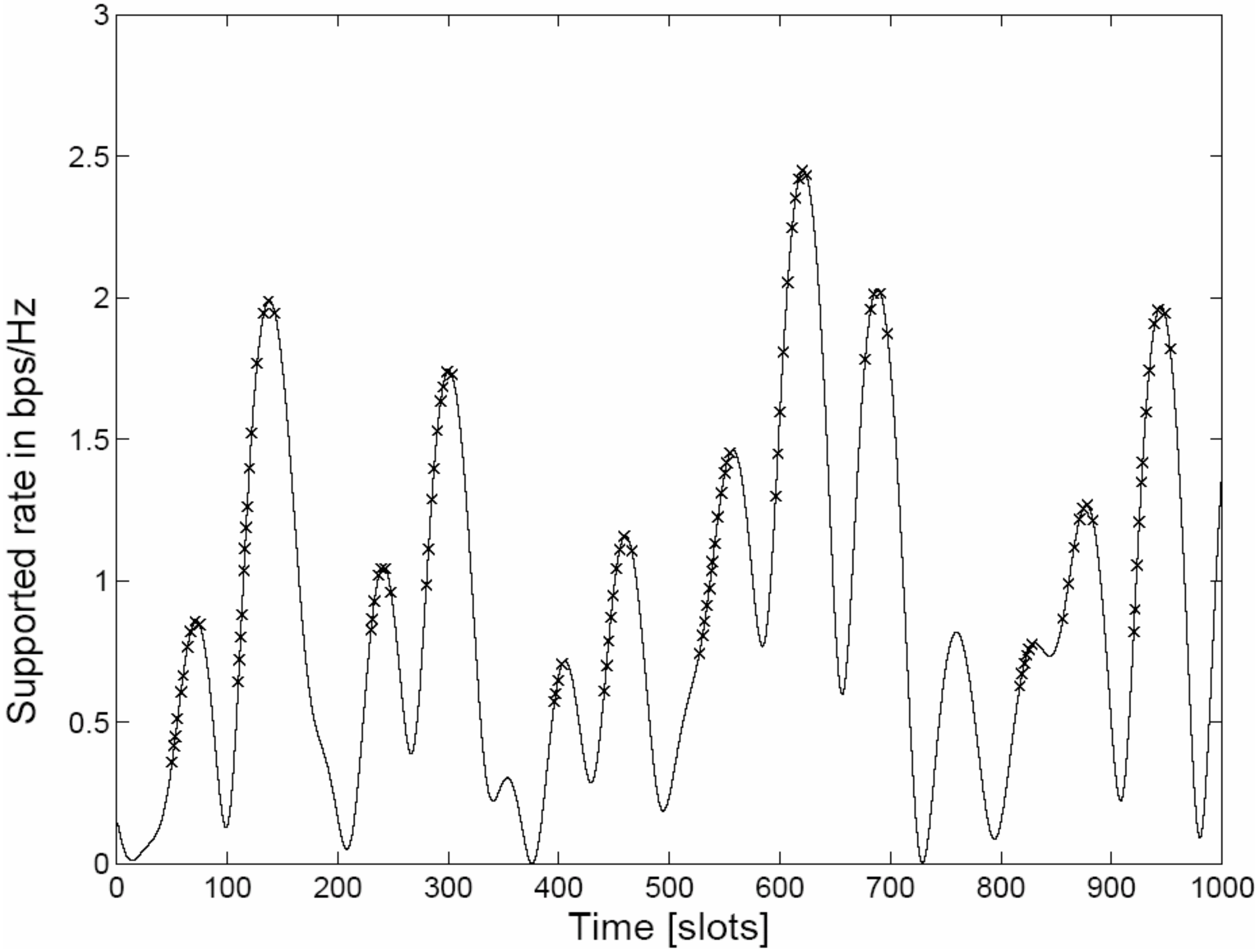


Simulation

- Ten users with equal channel statistics
- Average SNR 0dB
- Time slot Doppler frequency product 0.01
- Prediction 20 time slots ahead
- The supported rate and scheduling instances for one user

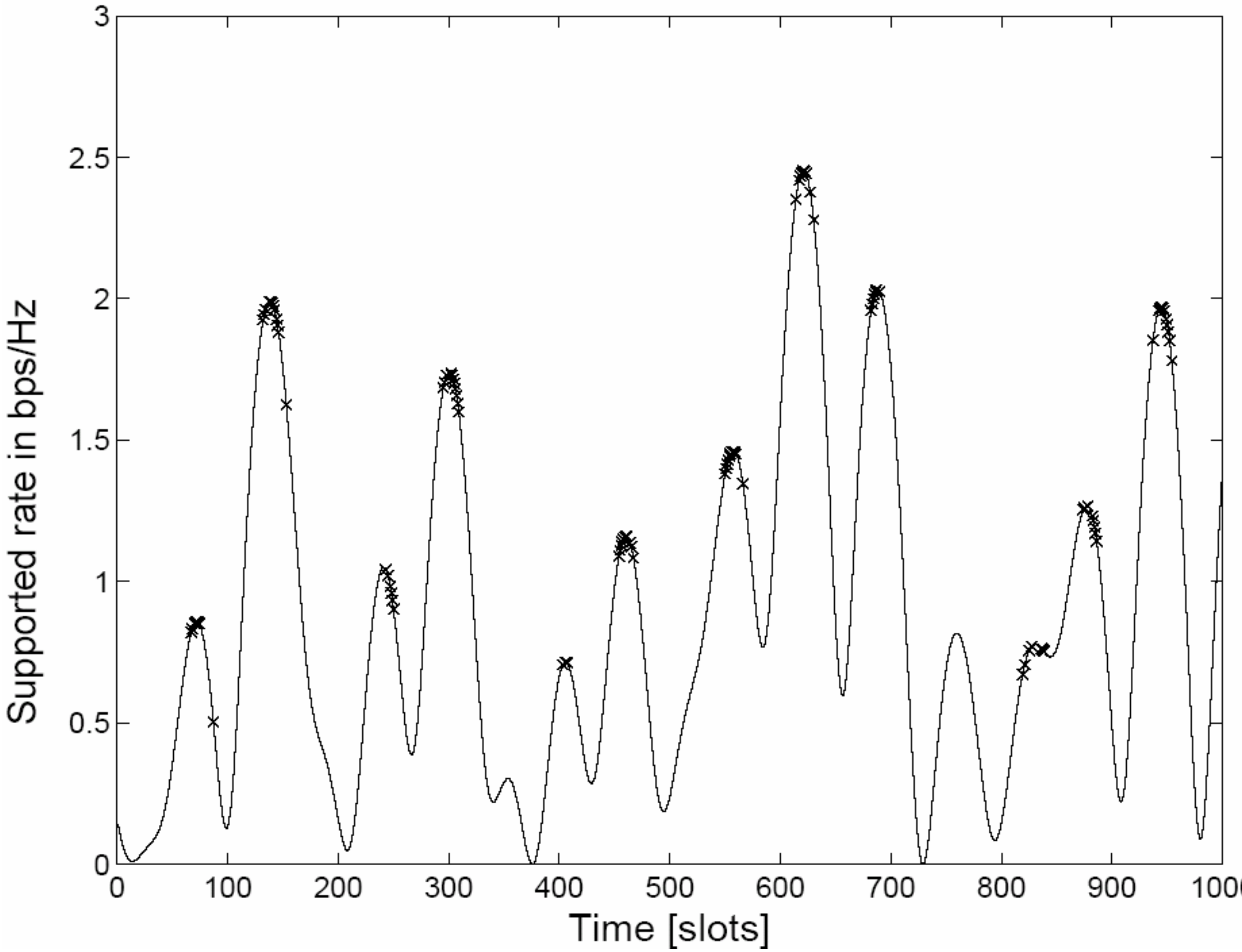


PFS



Supported rate in bps/Hz
Time [slots]
0
0.5
1
1.5
2
2.5
3
0
100
200
300
400
500
600
700
800
900
1000

P-PFS



Supported rate in bps/Hz
0
0.5
1
1.5
2
2.5
3
0
100
200
300
400
500
600
700
800
900
1000
Time [slots]

Proportional Fair Scheduling

- Pick the user with the highest ratio between rate and local accumulated throughput in the next time slot

$$i^*(k) = \arg \max_{i=1,..,N} \frac{R_i(k)}{T_i(k)}$$

- Optimized system utility function

  Sum of the log of the local throughputs

$$U(k) = \sum_{j=1}^{N} \log T_j(k)$$

- Exponential window for local accumulated throughput (time constant t_c)

$$T_i(k+1) = \begin{cases} \left(1 - \frac{1}{t_c}\right) T_i(k) + \frac{1}{t_c} R_i(k) & i = i^*(k) \\ \left(1 - \frac{1}{t_c}\right) T_i(k) & i \neq i^*(k) \end{cases}$$

Predictive Proportional Fair Scheduling (P-PFS)

- In time slot *k:*
  don't maximize $U(k+1)$, maximize $U(k+L)$
- Scheduling vector $\mathbf{i}(k)=(i_1,i_2,\ldots,i_L)$
- Schedule to maximize $U(k+L)$

$$\mathbf{i}^*(k) = \arg \max_{\mathbf{i} \in \mathcal{F}} \hat{U}(k+L|\mathbf{i})$$

- The estimated future system utility function $U(k+L)$, assuming user i_l is served in slot $k+l-1$ is $\hat{U}\left(k+L|(i_1, i_2 \ldots i_L)\right)$

Problems With Predictive Scheduling

- Future supported data rates are assumed known
  - Short range channel state predictions are good
  - Long rang predictions are quite poor
  - Don’t schedule too far
  - Don’t trust your schedule:

    Redo scheduling in each time step
- Full search of scheduling vectors to maximize a system utility function is computational demanding
  - Use possibly suboptimal iterative solutions

Cope With Prediction Uncertainty: Always Redo Scheduling!

k	$k+1$		$k+L-2$	$k+L-1$	Time step
$R_1(k\|k-1)$	$R_1(k+1\|k-1)$	…	$R_1(k+L-2\|k-1)$	$R_1(k+L-1\|k-1)$	Rate prediction quality decrease with increasing prediction range
$R_2(k\|k-1)$	$R_2(k+1\|k-1)$	…	$R_2(k+L-2\|k-1)$	$R_2(k+L-1\|k-1)$	
$R_N(k\|k-1)$	$R_N(k+1\|k-1)$	…	$R_N(k+L-2\|k-1)$	$R_N(k+L-1\|k-1)$	
$i_1(k)$	$i_2(k)$	…	$i_{L-1}(k)$	$i_L(k)$	Scheduling vector

Only effectuate the first component of the scheduling vector

New channel state information. **Update rate predictions**

Next time step

	$R_1(k+1\|k)$	$R_1(k+2\|k)$	…	$R_1(k+L-1\|k)$	$R_1(k+L\|k)$
	$R_2(k+1\|k)$	$R_2(k+2\|k)$	…	$R_2(k+L-1\|k)$	$R_2(k+L\|k)$
	$R_N(k+1\|k)$	$R_N(k+2\|k)$	…	$R_N(k+L-1\|k)$	$R_N(k+L\|k)$
Redo scheduling	$i_1(k+1)$	$i_2(k+1)$	…	$i_{L-1}(k+1)$	$i_L(k+1)$

Cope With Complexity: Iterative Search!

$i_1(k\text{-}1)$	$i_2(k\text{-}1)$	…	$i_{L\text{-}1}(k\text{-}1)$	$i_L(k\text{-}1)$	= $\mathbf{i}(k\text{-}1)$ Previous scheduling vector
$i_2(k\text{-}1)$	$i_3(k\text{-}1)$	…	$i_L(k\text{-}1)$	1	= $\mathbf{i}^0(k)$ Initialization
$i_2(k\text{-}1)$	$i_3(k\text{-}1)$	…	$i_L(k\text{-}1)$	$i_L^1(k)$	= $\mathbf{i}^1(k)$ First iteration
$i_2(k\text{-}1)$	$i_3(k\text{-}1)$	…	$i_{L-1}^2(k)$	$i_L^1(k)$	= $\mathbf{i}^2(k)$ Second iteration
		⋮			
$i_1^L(k)$	$i_2^{L-1}(k)$	…	$i_{L-1}^2(k)$	$i_L^1(k)$	= $\mathbf{i}^L(k)$ L:th iteration
$i_1^L(k)$	$i_2^{L-1}(k)$	…	$i_{L-1}^2(k)$	$i_L^{L+1}(k)$	= $\mathbf{i}^{L+1}(k)$ $L+1$:th iteration

Keep iterating until it converges

Each iteration one component of the vector is recomputed, all the others are held fixed

$$i_l^{n+1}(k) = \arg\max_{i=1,..,N} \hat{U}\left(k+L \,|\, \mathbf{i}^n(k) \xleftarrow{l} i\right)$$

Some Comments on the Algorithm

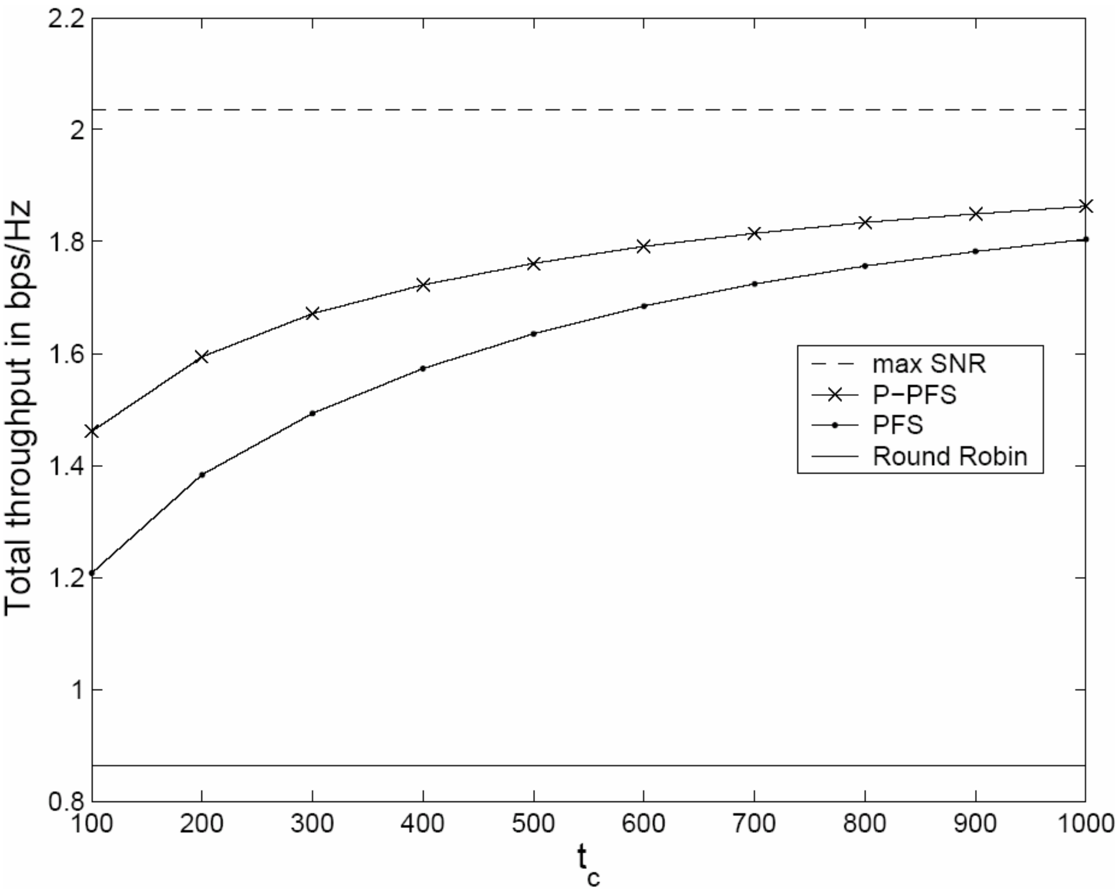
In the proposed frame work

- Any rate predictor can be used
  - It should be conservative
- Any utility function U can be used
  - Here a generalization of PFS leads to maximizing $U(k+L)$.
  - It is feasible to redefine U to instead maximize $U(k+1)$ taking past and future rates into account
- The iterations converge fast
  - A small amount of new channel state information is introduced at each time step
  - The initial scheduling vector is based on a vector obtaining a maximum in the previous time step

Prediction Leads to Higher Throughput

Simulation

- 15 users
- Equal channel statistics
- Average SNR 0dB
- Time slot Doppler frequency product 0.01
- Prediction range: 10 slots



How to Measure Fairness

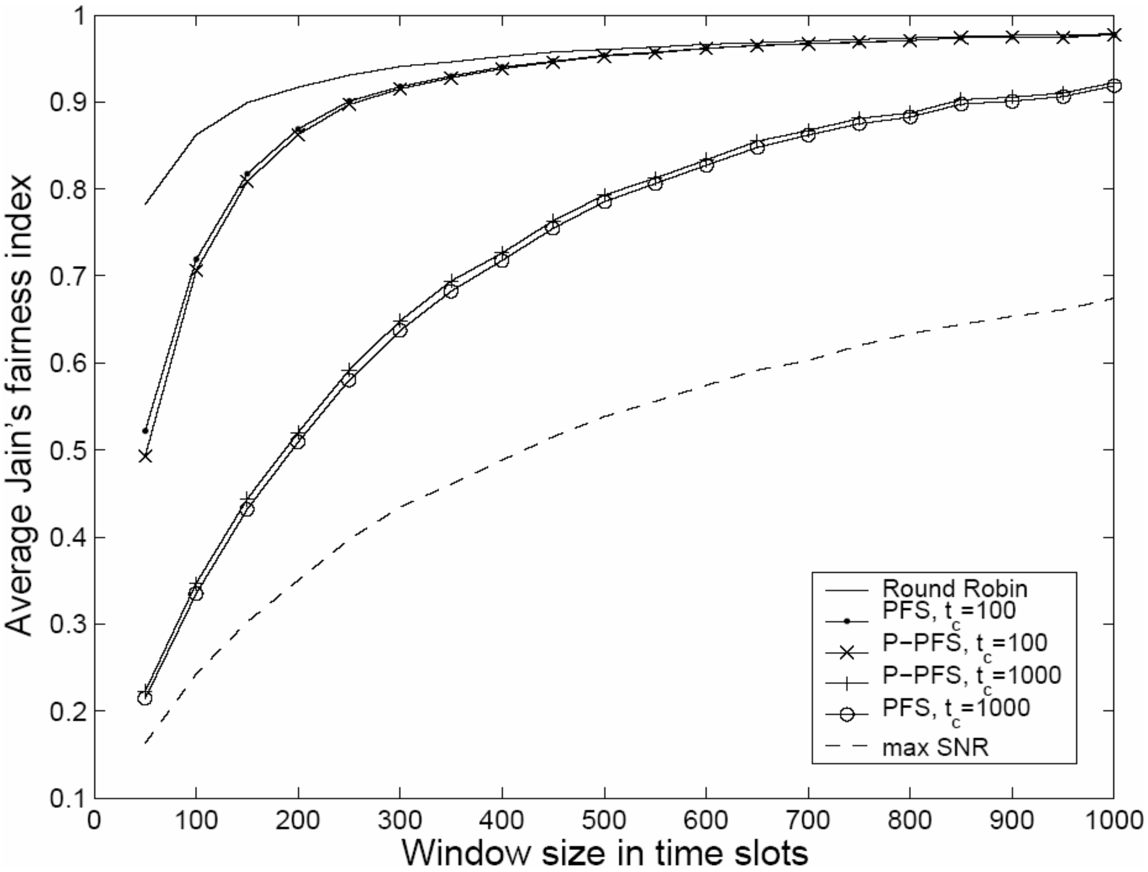
- Jain's fairness index
- Measures spread of the users average throughput (rectangular window)
- $J=1$ absolute fairness
- $J=1/N$ totally unfair (all resources to one user)
- N is the number of users

$$J = \frac{\left(\sum_{i=1}^{N} T_i\right)^2}{N \sum_{i=1}^{N} T_i^2}$$

Exploiting predictions doesn't compromise fairness

Simulation

- 15 users
- Equal channel statistics
- Average SNR 0dB
- Time slot Doppler frequency product 0.01
- Prediction range: 10 slots



Conclusion

- Introduced a wireless scheduling algorithm
- Exploiting fading predictions in a robust manner
- Reasonable increase in complexity
- Increased throughput without compromising fairness

- This activity will be continued within the MoPSAR project