Journal of Applied Computer Science Methods

Published by University of Social Sciences

ACSM

Volume 6 Number 1 2014

University of Social Sciences, IT Institute

Społeczna Akademia Nauk
Łódź

INTERNATIONAL JOURNAL OF APPLIED COMPUTER SCIENCE METHODS (JACSM)
is a semi-annual periodical published by the University of Social Sciences (SAN)
in Lodz, Poland.

PUBLISHING AND EDITORIAL OFFICE:
University of Social Sciences (SAN)
Information Technology Institute (ITI)
Sienkiewicza 9
90-113 Lodz
Tel.: +48 42 6646654
Fax.: +48 42 6366251
E-mail: acsm@swspiz.pl
URL: http://acsm.swspiz.pl

Print: Mazowieckie Centrum Poligrafii, ul. Duża 1, 05-270 Marki, www.c-p.com.pl, biuro@c-p.com.pl

Copyright © 2014 University of Social Sciences, Lodz, Poland. All rights reserved.

AIMS AND SCOPE:

The **International Journal of Applied Computer Science Methods is** a semi-annual, refereed periodical, publishes articles describing recent contributions in theory, practice and applications of computer science. The broad scope of the journal includes, but is not limited to, the following subject areas:

Knowledge Engineering and Information Management: *Knowledge Processing, Knowledge Representation, Data Mining, Machine Learning, Knowledge-based Systems, Knowledge Elicitation, Knowledge Acquisition, E-learning, Web-intelligence, Collective Intelligence, Language Processing, Approximate Reasoning, Information Archive and Processing, Distributed Information Systems.*

Intelligent Systems: *Intelligent Database Systems, Expert Systems, Decision Support Systems, Intelligent Agent Systems, Artificial Neural Networks, Fuzzy Sets and Systems, Evolutionary Methods and Systems, Hybrid Intelligent Systems, Cognitive Systems, Intelligent Systems and Internet, Complex Adaptive Systems.*

Image Understanding and Processing: *Computer Vision, Image Processing, Computer Graphics, Pattern Recognition, Virtual Reality, Multimedia Systems.*

Computer Modeling, Simulation and Soft Computing: *Applied Computer Modeling and Simulation, Intelligent Computing and Applications, Soft Computing Methods, Intelligent Data Analysis, Parallel Computing, Engineering Algorithms.*

Applied Computer Methods and Computer Technology: *Programming Technology, Database Systems, Computer Networks Technology, Human-computer Interface, Computer Hardware Engineering, Internet Technology, Biocybernetics.*

DISTRIBUTION:

Apart from the standard way of distribution (in the conventional paper format), on-line dissemination of the JACSM is possible for interested readers.

Contents

DE GRUYTER
OPEN

JACSM 2014, Vol. 6, No. 1, pp. 5 - 26
DOI 10.2478/jacsm-2014-0001

MINIMUM UTILIZATION OF ELECTROMAGNETIC (2G, 3G, 4G) SPECTRUM IN SEAMLESS MOBILITY BASED ON VARIOUS ESTIMATION METHODS

A. Jayanthila Devi[1], Dr. G. M. Kadhar Nawaz[2]

[1]Assistant Professor/MCA,Jain University, Bangalore
jayanthilamca@gmail.com

[2]Director/M CA,Sona College of Technology, Salem
nawazse@yahoo.co.in

Abstract

Wireless Cellular Seamless Mobility is the most important feature of a wireless cellular communication system. Basically cell phones are used for communication purpose and therefore good cell coverage is needed to make and receive calls. Usually, continuous service is achieved by supporting handover from one cell to another. Handover is the process of changing the channel associated with the current connection while a call is in progress. It is often initiated either by crossing a cell boundary or by deterioration in quality of the signal in the current channel. Mobile IP, too, has to support redirection of data to a new foreign agent after the change of network access. In this case the additional delay caused by the redirection. The shorter the interruption of the service is –up to the ideal case of no service interruption. Minimizing the Delay based on some prior estimation methods in handover is better for quality of seamless connectivity anytime; anywhere across communication networks. All mobile phone system supports the seamless handover between base stations. To avoid the additional delay and wastage of Electromagnetic Spectrum, Mobile IP may use Optimizations such as rerouting of the whole packet in the flow of efficiently transmission of data with short interrupted service by Time series based Spectrum Estimation Technique. Due to the utilization of seamless handover mechanism it reduces the unwanted usage of Spectrum.

Key words: Estimation, Frequency, Handover, Seamless, Spectrum, Time

1 Introduction

The Electromagnetic Spectrum is the range of frequencies of possible electromagnetic radiation. The Spectrum ranges from 0 Hertz up to 2.4x1023 Hertz. The exact wavelength limits of the Spectrum are unknown however it is widely believed that the short wavelength limit is equal to the Planck

Length (1.616x10-35m) and the long wavelength limit is the length of the Universe.[1]. The frequencies of electromagnetic radiation can be calculated by dividing the speed of light by the wavelength of the electromagnetic radiation. Regions of the Electromagnetic Spectrum have been named by scientists to provide an easier way to remember and refer to the ranges; however, in reality neighbouring types of electromagnetic energy often overlap. The goal of Spectrum Estimation is to describe the distribution (over frequency) of the contained in a signal, based on a finite set of data. Estimation of spectrum is useful in a variety of applications, including the detection of signals buried in wideband noise.

2 Spectrum Estimation

The Spectrum Estimation of a stationary random process xn is mathematically related to the autocorrelation sequence by the discrete-time Simulation. In terms of normalized frequency, this is given by

wavelength* × *frequency* = *the speed of light
or
$$\lambda \times f = c$$

In this equation, the Greek letter "lambda" (λ) is used as shorthand for the wavelength and the fancy "f" (*f*) is used to represent the frequency; "c" is the speed of light (186,000 miles per second or 300 million meters per second). Since the speed of light is constant, the wavelength and frequency are limited; if one is big the other has to be small.[2]. That is why large (high) frequencies correspond to small wavelengths and large wavelengths correspond to small (low) frequencies.To convert from frequency (f) to wavelength (8) and vice versa, recall that f = c/8, or 8 = c/f; where c = speed of light. Rules follow:

Metric:

Wavelength in cm = 30 / frequency in GHz
For example: at 10 GHz, the wavelength = 30/10 = 3 cm
Wavelength in ft = 1 / frequency in GHz
For example: at 10 GHz, the wavelength = 1/10 = 0.1 ft

$$P_{xx}(\omega) = \frac{1}{2\pi} \sum_{m=-\infty}^{\infty} R_{xx}(m) e^{-j\omega m}$$

This can be written as a function of physical frequency f (e.g., in hertz) by using the relation $\omega = 2\pi f/f_s$, where f_s is the sampling frequency.

$$P_{xx}(f) = \frac{1}{f_s} \sum_{m=-\infty}^{\infty} R_{xx}(m) e^{-j2\pi nf/f_s}$$

The correlation sequence can be derived from the SE by use of the inverse discrete-time Simulation:

$$R_{xx}(m) = \int_{-\pi}^{\pi} P_{xx}(\omega) e^{j\omega m} d\omega = \int_{-f_s/2}^{f_s/2} P_{xx}(f) e^{j2\pi nf/f_s} df$$

The average of the sequence x_n over the entire interval is represented by

$$R_{xx}(0) = \int_{-\pi}^{\pi} P_{xx}(\omega) d\omega = \int_{-f_s/2}^{f_s/2} P_{xx}(f) df$$

The average of a signal over a particular frequency band $[\omega_1, \omega_2]$, $0 \le \omega_1 \le \omega_2 \le \pi$, can be found by integrating the SE over that band:

$$\overline{P}_{[\omega_1,\omega_2]} = \int_{\omega_1}^{\omega_2} P_{xx}(\omega) d\omega = \int_{-\omega_2}^{-\omega_1} P_{xx}(\omega) d\omega$$

You can see from the above expression that $P_{xx}(\omega)$ represents the content of a signal in an *infinitesimal* frequency band, which is why it is called the spectrum *Estimation*. The units of the SE are (e.g., watts) per unit of frequency. In the case of $P_{xx}(\omega)$, this is watts/radian/sample or simply watts/radian. In the case of $P_{xx}(f)$, the units are watts/hertz. Integration of the SE with respect to frequency yields units of watts, as expected for the average .For real–valued signals, the SE is symmetric about DC, and thus $P_{xx}(\omega)$ for $0 \le \omega \le \pi$ is sufficient to completely characterize the SE. However, to obtain the average over the entire Nyquist interval, it is necessary to introduce the concept of the *one-sided* SE.The one-sided SE is given by

$$P_{\text{onesided}}(\omega) = \begin{cases} 0 & -\pi \le \omega < 0 \\ 2P_{xx}(\omega) & 0 \le \omega \le \pi \end{cases}$$

The average of a signal over the frequency band, $[\omega_1, \omega_2]$ with $0 \le \omega_1 \le \omega_2 \le \pi$, can be computed using the one-sided SE as

$$\bar{P}_{[\omega_1,\omega_2]} = \int_{\omega_1}^{\omega_2} P_{\text{onesided}}(\omega)d\omega$$

3 Spectrum Estimation Method

The various methods of estimation available are categorized as follows: In general terms, one way of estimating the SE of a process is to simply find the discrete-time series of the samples of the process (usually done on a grid with an FFT) and appropriately scale the magnitude squared of the result. This estimate is called the *timeseries*.The time series estimate of the SE of a length-L signal $x_L[n]$ is

$$P_{xx}(f) = \frac{1}{LF_s}\left|\sum_{n=0}^{L-1} x_L(n)e^{-j2\pi fn/F_s}\right|^2$$

Where F_s is the sampling frequency. In practice, the actual computation of $P_{xx}(f)$ can be performed only at a finite number of frequency points, and usually employs an FFT. Most implementations of the time series method compute the *N*-point SE estimate at the frequencies.

$$f_k = \frac{kF_s}{N} \quad k = 0,1,\ldots,N-1$$

In some cases, the computation of the time series via an FFT algorithm is more efficient if the number of frequencies is a of two. Therefore it is not uncommon to pad the input signal with zeros to extend its length to a of two. As an example of the time series, consider the following 1001-element signal xn, which consists of two sinusoids plus noise:

```
fs = 1000;              % Sampling frequency
t = (0:fs)/fs;          % One second worth of samples
A = [1 2];              % Sinusoid amplitudes (row vector)
f = [150;140];          % Sinusoid frequencies (column vector)
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
```

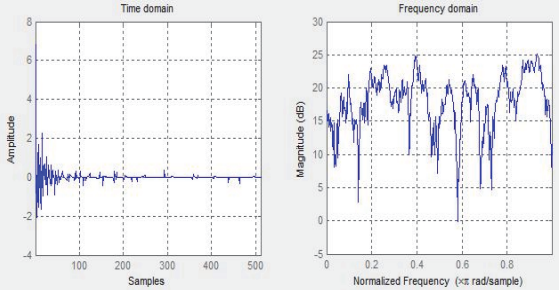


Figure 1. Time and Frequency Domian Estimation samples

The timeseries estimate of the SE can be computed using timeseries. In this case, the data vector is multiplied by a Hamming window to produce a modified timeseries.[3].

[Pxx,F] = time series(xn,hamming(length(xn)),length(xn),fs);
*plot(F,10*log10(Pxx))*
xlabel('Hz'); ylabel('dB');
title('Modified Timeseries Spectrum Estimation Estimate');

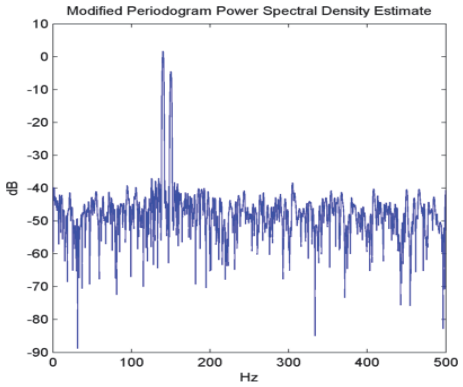


Figure 2. Performance of the Time series

The following sections discuss the performance of the timeseries with regard to the issues of leakage, resolution, bias, and variance.

3.1 S*pectrum* Leakage:

Consider the SE of a finite-length (length L) signal $x_L[n]$, as discussed in the Timeseries section. It is frequently useful to interpret $x_L[n]$ as the result of

multiplying an infinite signal, $x[n]$, by a finite-length rectangular window, $w_R[n]$:

$$x_L[n] = x[n]w_R[n].$$

Because multiplication in the time domain corresponds to convolution in the frequency domain, the expected value of the timeseries in the frequency domain is:

$$E\{P_{xx}(f)\} = \frac{1}{Fs}\int_{-Fs/2}^{Fs/2} \frac{\sin^2(L\pi(f-f')/Fs)}{L\sin^2(\pi(f-f')/Fs)} P_{xx}(f')df',$$

showing that the expected value of the timeseries is the convolution of the true SE with the square of the Dirichlet kernel.The effect of the convolution is best understood for sinusoidal data. Suppose that $x[n]$ is composed of a sum of M complex sinusoids:

$$x(n) = \sum_{k=1}^{N} A_k e^{j\omega_k n}$$

Its is

$$X(\omega) = \sum_{k=1}^{N} A_k \delta(\omega - \omega_k)$$

which for a finite-length sequence becomes

$$X(\omega) = \int_{-\pi}^{\pi} (\sum_{k=1}^{N} A_k \delta(\ -\omega_k)) W_R(\omega - \ )d \ = = \ \backslash \ = \ '$$

The preceding equation is equal to:

$$X(\omega) = \sum_{k=1}^{N} A_k W_R(\omega - \omega_k)$$

So in the of the finite-length signal, the Dirac deltas have been replaced by terms of the form $W_R(\omega-\omega_k)$, which corresponds to the frequency response of a rectangular window centered on the frequency ω_k.The frequency response of a rectangular window has the shape of a sinc signal, as shown below.

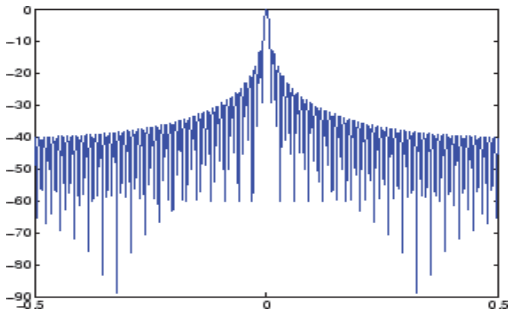


Figure 3. Variance of the Timeseries Spectrum Leakage.

The plot displays a main lobe and several side lobes, the largest of which is approximately 13.5 dB below the main lobe peak. These lobes account for the effect known as *spectrum leakage*. While the infinite-length signal has its concentrated exactly at the discrete frequency points f_k, the windowed (or truncated) signal has a continuum of "leaked" around the discrete frequency points f_k.Because the frequency response of a short rectangular window is a much poorer approximation to the Dirac delta function than that of a longer window, spectrum leakage is especially evident when data records are short. Consider the following sequence of 100 samples:

```
fs = 1000;              % Sampling frequency
t = (0:fs/10)/fs;       % One-tenth second worth of samples
A = [1 2];              % Sinusoid amplitudes
f = [150;140];          % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
[Pxx,F] = timeseries(xn,rectwin(length(xn)),length(xn),fs);
plot(F,10*log10(Pxx))
```

It is important to note that the effect of spectrum leakage is contingent solely on the length of the data record. It is *not* a consequence of the fact that the time series is computed at a finite number of frequency samples.

3.2 Resolution:

Resolution refers to the ability to discriminate spectrum features, and is a key concept on the analysis of spectrum estimator performance.In order to resolve two sinusoids that are relatively close together in frequency, it is necessary for the difference between the two frequencies to be greater than the width of the main lobe of the leaked spectra for either one of these sinusoids. The mainlobe width is defined to be the width of the main lobe at the point

where the is half the peak main lobe (i.e., 3 dB width). This width is approximately equal to f_s / L.[4].

In other words, for two sinusoids of frequencies f_1 and f_2, the resolvability condition requires that

$$(f_2 - f_1) > \frac{Fs}{L}$$

In the example above, where two sinusoids are separated by only 10 Hz, the data record must be greater than 100 samples to allow resolution of two distinct sinusoids by a timeseries.Consider a case where this criterion is not met, as for the sequence of 67 samples below:

```
fs = 1000;              % Sampling frequency
t = (0:fs/15)./fs;      % 67 samples
A = [1 2];              % Sinusoid amplitudes
f = [150;140];          % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
[Pxx,F] = time series(xn,rectwin(length(xn),length(xn,fs);
plot(F,10*log10(Pxx))
```

The above discussion about resolution did not consider the effects of noise since the signal-to-noise ratio (SNR) has been relatively high thus far. When the SNR is low, true spectrum features are much harder to distinguish, and noise artifacts appear in spectrum estimates based on the timeseries.The example below illustrates this:

```
fs = 1000;              % Sampling frequency
t = (0:fs/10)./fs;      % One-tenth second worth of samples
A = [1 2];              % Sinusoid amplitudes
f = [150;140];          % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 2*randn(size(t));
[Pxx,F] = timeseries (xn,rectwin(length (xn)),length(xn),fs);
plot(F,10*log10(Pxx))
```

3.3 Bias of the Timeseries:

The timeseries is a biased estimator of the SE. Its expected value was previously shown to be:

$$E[P_{xx}(f)] = \frac{1}{Fs}\int_{-Fs/2}^{Fs/2} \frac{\sin^2(L\pi(f-f')/Fs)}{L\sin^2(\pi(f-f')/Fs)} P_{xx}(f')df',$$

The timeseries is asymptotically unbiased, which is evident from the earlier observation that as the data record length tends to infinity, the frequency response of the rectangular window more closely approximates the Dirac delta function. However, in some cases the timeseries is a poor estimator of the SE even when the data record is long. This is due to the variance of the timeseries, as explained below.

3.4 Variance of the Timeseries:

The variance of the timeseries can be shown to be:

$$\mathrm{Var}(P_{xx}(f)) = \begin{cases} P_{xx}^2(f) & 0 < f < \mathrm{Fs}/2 \\ 2P_{xx}^2(f) & f = 0 \quad \text{or } f = Fs/2 \end{cases}$$

which indicates that the variance does not tend to zero as the data length L tends to infinity. In statistical terms, the timeseries is not a consistent estimator of the SE. Nevertheless, the timeseries can be a useful tool for spectrum estimation in situations where the SNR is high, and especially if the data record is long.

4 The Modified Time series:

The *modified time series* windows the time-domain signals prior to computing the DFT in order to smooth the edges of the signal. This has the effect of reducing the height of the sidelobes or spectrum leakage. This phenomenon gives rise to the interpretation of sidelobes as spurious frequencies introduced into the signal by the abrupt truncation that occurs when a rectangular window is used. For nonrectangular windows, the end points of the truncated signal are attenuated smoothly, and hence the spurious frequencies introduced are much less severe. On the other hand, nonrectangular windows also broaden the mainlobe, which results in a reduction of resolution.[5].

Timeseries allows you to compute a modified timeseries by specifying the window to be used on the data.

For example, compare a default rectangular window and a Hamming window:

```
fs = 1000;              % Sampling frequency
t = (0:fs/10)./fs;      % One-tenth second worth of samples
A = [1 2];              % Sinusoid amplitudes
f = [150;140];          % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
[Pxx,F] = timeseries(xn,rectwin(length(xn)),length(xn),fs);
```

*plot(F,10*log10(Pxx))*
[Pxx,F] = timeseries(xn,hamming(length(xn)),length(xn),fs);
*plot(F,10*log10(Pxx))*

You can verify that although the sidelobes are much less evident in the Hamming-windowed timeseries, the two main peaks are wider. In fact, the 3 dB width of the mainlobe corresponding to a Hamming window is approximately twice that of a rectangular window. Hence, for a fixed data length, the SE resolution attainable with a Hamming window is approximately half that attainable with a rectangular window.The competing interests of mainlobe width and sidelobe height can be resolved to some extent by using variable windows such as the window.Nonrectangular windowing affects the average of a signal because some of the time samples are attenuated when multiplied by the window. To compensate for this, timeseries and p normalize the window to have an average of unity.This ensures that the measured average is generally independent of window choice. If the frequency components are not well resolved by the SE estimators, the window choice does affect the average. The modified timeseries estimate of the SE is

$$\hat{P}_{xx}(f) = \frac{|X(f)|^2}{FsLU}$$

where U is the window normalization constant.

$$U = \frac{1}{L}\sum_{n=0}^{N-1} |w(n)|^2 .$$

For large values of L, U tends to become independent of window length. The addition of U as a normalization constant ensures that the modified timeseries is asymptotically unbiased.

4.1 Timeseries

An improved estimator of the SE is the one proposed by [8]. The method consists of dividing the time series data into (possibly overlapping) segments, computing a modified timeseries of each segment, and then averaging the SE estimates. The result is's SE estimate. Timeseries is implemented in the toolbox by or function. The averaging of modified timeservers tends to decrease the variance of the estimate relative to a single timeseries estimate of the entire data record. Although overlap between segments introduces redundant information, this effect is diminished by the use of a nonrectangular window, which reduces the importance or *weight* given to the end samples of segments (the samples that overlap).

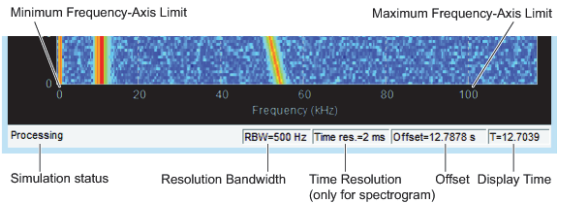


Figure 4. Minimum and Maximum Frequency Axis Limit

Hz per unit time, or:
Time in pass band = RBW
Span/ST = (RBW)(ST)
Span Where
RBW = resolution bandwidth and
ST = sweep time.
On the other hand, the rise time of a filter
is inversely proportional to its bandwidth,
and if we include a constant of proportionality, k, then:
Rise time = k
RBW
If we make the terms equal and solve for
sweep time, we have:
k = (RBW)(ST)
Span
or ST = k (Span)
RBW2

Entering the values from our example into the equation, we get:
2.12H(4 kHz) = –3.01 dB x = –24.1 dB .At an offset of 4 kHz, the 3-kHz digital filter is down –24.1 dB compared to the analog filter which was only down –14.8 dB. Because of its superior selectivity, the digital filter can resolve more closely spaced signals.

The general expression used to calculate
the maximum mismatch error in dB is:
Error (dB) = –20 log[1 ± |(ρanalyzer)(ρsource[|(where ρ is the reflection coefficient_

However, as mentioned above, the combined use of short data records and nonrectangular windows results in reduced resolution of the estimator. In summary, there is a trade-off between variance reduction and resolution. One

can manipulate the parameters in Timeseries to obtain improved estimates relative to the timeseries, especially when the SNR is low. This is illustrated in the following example.

Consider an original signal consisting of 1000 samples:

```
fs = 1000;            % Sampling frequency
t = (0:1*fs)./fs;    % 301 samples
A = [2 8];            % Sinusoid amplitudes (row vector)
f = [150;140];        % Sinusoid frequencies (column vector)
xn = A*sin(2*pi*f*t) + 5*randn(size(t));
[Pxx,F] = timeseries(xn,rectwin(length(xn)),length(xn),fs);
plot(F,10*log10(Pxx))
```

We can obtain 's spectrum estimate for 3 segments with 50% overlap using a Hamming window.

```
[Pxx,F] = p(xn,hamming(150),75,150,fs);
plot(F,10*log10(Pxx)); xlabel('Hz'); ylabel('dB');
title('''s Overlapped Segment Averaging SE Estimate');
```

In the timeseries above, noise and the leakage make one of the sinusoids essentially indistinguishable from the artificial peaks. In contrast, although the SE produced by Timeseries has wider peaks, you can still distinguish the two sinusoids, which stand out from the "noise floor."However, if we try to reduce the variance further, the loss of resolution causes one of the sinusoids to be lost altogether:

```
[Pxx,F] = p(xn,rectwin(100),75,512,Fs);
plot(F,10*log10(Pxx))
```

4.2 Bias and Normalization in Timeseries:

Timeseries yields a biased estimator of the SE. The expected value of the SE estimate is:

$$E\{P_{\text{Welch}}(f)\} = \frac{1}{FsLU}\int_{-Fs/2}^{F/2} |W(f-f')|^2 \, P_{xx}(f')df'$$

Where *L* is the length of the data segments, *U* is the same normalization constant present in the definition of the modified timeseries, and *W(f)* is the Fourier transform of the window function. As is the case for all timeservers,

estimator is asymptotically unbiased. For a fixed length data record, the bias of's estimate is larger than that of the timeseries because the length of the segments is less than the length of the entire data sample.

The variance of estimator is difficult to compute because it depends on both the window used and the amount of overlap between segments. Basically, the variance is inversely proportional to the number of segments whose modified timeseriess are being averaged.

```
 fs = 1000;            % Sampling frequency
 t = (0:fs)/fs;        % One second worth of samples
A = [1 2];             % Sinusoid amplitudes
f = [150;140];         % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
[Pxx1,F1] = pmtm(xn,4,fs);
plot(F1,10*log10(Pxx1))
```

By lowering the time-bandwidth product, you can increase the resolution at the expense of larger variance:

```
[Pxx2,F2] = pmtm(xn,1.5,fs);
plot(F2,10*log10(Pxx2))
```

This method is more computationally expensive than Timeseries due to the cost of computing the discrete prolate spheroidal sequences. For long data series (10,000 points or more), it is useful to compute the DPSSs once and save them in a MAT-file. dpsssave,dpssload, dpsSEir, and dpssclear are provided to keep a database of saved DPSSs in the MAT-file dpss.mat.

Cross-Spectrum Estimation Function

The SE is a special case of the *cross spectrum Estimation* (*CSE*) function, defined between two signals x_n and y_n as

$$P_{xy}(\omega) = \frac{1}{2\pi} \sum_{m=-\infty}^{\infty} R_{xy}(m) e^{-j\omega m}$$

As is the case for the correlation and covariance sequences, the toolbox *estimates* the SE and CSE because signal lengths are finite.

To estimate the cross-spectrum Estimation of two equal length signals x and y using Timeseries, the cSE function forms the timeseries as the product of the FFT of x and the conjugate of the FFT of y. Unlike the real-valued SE, the CSE is a complex function.cSE handles the sectioning and windowing of x and y in the same way as the p function:

Sxy = cSE(x,y,nwin,noverlap,nfft,fs)

4.3 Transfer Function Estimate:

One application of Timeseries is nonparametric system identification. Assume that H is a linear, time invariant system, and $x(n)$ and$y(n)$ are the input to and output of H, respectively. Then the of $x(n)$ is related to the CSE of $x(n)$ and $y(n)$ by

$$P_{yx}(\omega) = H(\omega)P_{xx}(\omega)$$

An estimate of the transfer function between $x(n)$ and $y(n)$ is

$$\hat{H}(\omega) = \frac{\hat{P}_{yx}(\omega)}{\hat{P}_{xx}(\omega)}$$

This method estimates both magnitude and phase information. The tfestimate function uses Timeseries to compute the CSE and , and then forms their quotient for the transfer function estimate. Use tfestimate the same way that you use thecSE function.

Filter the signal xn with an FIR filter, then plot the actual magnitude response and the estimated response:

```
h = ones(1,10)/10;                % Moving-average filter
yn = filter(h,1,xn);
[HEST,f] = tfestimate(xn,yn,256,128,256,fs);
H = freqz(h,1,f,fs);
subplot(2,1,1); plot(f,abs(H));
title('Actual Transfer Function Magnitude');
subplot(2,1,2); plot(f,abs(HEST));
title('Transfer Function Magnitude Estimate');
xlabel('Frequency (Hz)');
```

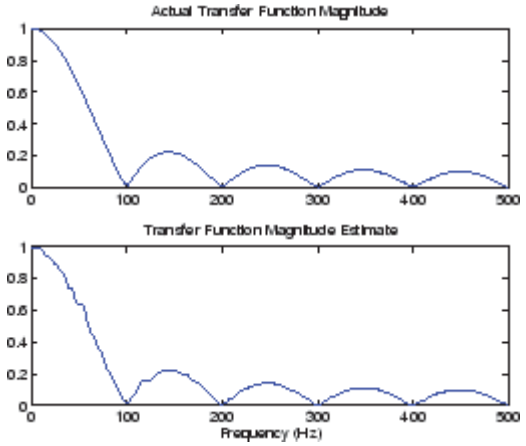


Figure 5. Frequency Function Estimation

4.4 Coherence Function:

The magnitude-squared coherence between two signals $x(n)$ and $y(n)$ is

$$C_{xy}(\omega) = \frac{|P_{xy}(\omega)|^2}{P_{xx}(\omega)P_{yy}(\omega)}$$

This quotient is a real number between 0 and 1 that measures the correlation between $x(n)$ and $y(n)$ at the frequency ω.

The cohere function takes sequences x and y, computes their spectra and CSE, and returns the quotient of the magnitude squared of the CSE and the product of the spectra. Its options and operation are similar to the cSE and tfestimate functions.The coherence function of xn and the filter output yn versus frequency is mscohere(xn,yn,256,128,256,fs).

If the input sequence length nfft, window length window, and the number of overlapping data points in a window numoverlap, are such that mscohere operates on only a single record, the function returns all ones. This is because the coherence function for linearly dependent data is one.All AR methods yield a SE estimate given by

$$\hat{P}(f) = \frac{1}{Fs} \frac{p}{\left|1 - \sum_{k=1}^{p} \hat{a}_p(k) e^{-j2\pi kf/Fs}\right|^2}$$

The different AR methods estimate the parameters slightly differently, yielding different SE estimates. The following table provides a summary of the different AR methods.*Method* of spectrum estimation computes the AR parameters by solving the following linear system, which give the equations in matrix form:

$$\begin{pmatrix} r(0) & r^*(1) & \dots & r^*(p-1) \\ r(1) & r(0) & \dots & r^*(p-2) \\ r(p-1) & r(p-2) & \dots & r(0) \end{pmatrix} \begin{pmatrix} a(1) \\ a(2) \\ \dots \\ a(p) \end{pmatrix} = \begin{pmatrix} r(1) \\ r(2) \\ \dots \\ r(p) \end{pmatrix}$$

In practice, the biased estimate of the autocorrelation is used for the unknown true autocorrelation.The method produces the same results as a maximum entropy estimator.The use of a biased estimate of the autocorrelation function ensures that the autocorrelation matrix above is positive definite. Hence, the matrix is invertible and a solution is guaranteed to exist. Moreover, the AR parameters thus computed always result in a stable all-pole model.

The equations can be solved efficiently via Levinson's algorithm, which takes advantage of the Hermitian Toeplitz structure of the autocorrelation matrix..

For example, compare the of a speech signal using Timeseries and the AR method:

```
load mtlb
[Pxx,F] = p(mtlb,hamming(256),128,1024,Fs);
plot(F,10*log10(Pxx))

ORDER = 14;
[Pxx,F] = p(mtlb,ORDER,1024,fs);
plot(F,10*log10(Pxx))
```

The smoother than the timeseries because of the simple underlying all-pole model.The accuracy of the Burg method is lower for high-order models, long data records, and high signal-to-noise ratios (which can cause *line splitting*, or the generation of extraneous peaks in the estimate). The spectrum Estimation estimate computed by the Burg method is also susceptible to frequency shifts (relative to the true frequency) resulting from the initial phase of noisy sinusoidal signals. This effect is magnified when analyzing short data sequences.

Compare the of the speech signal generated method. They are very similar for large signal lengths:

```
load mtlb
ORDER = 14;
[Pburg,F] = pburg(mtlb(1:512),ORDER,1024,fs);
plot(F,10*log10(Pburg))
[Pxx,F] = p(mtlb(1:512),ORDER,1024,fs);
plot(F,10*log10(Pxx))
```

Compare the of a noisy signal computed using the method and the method:

```
fs = 1000;              % Sampling frequency
t = (0:fs)/fs;          % One second worth of samples
A = [1 2];              % Sinusoid amplitudes
f = [150;140];          % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
p(xn,hamming(256),128,1024,fs);
pburg(xn,14,1024,fs)
```

Note that, as the model order for the Burg method is reduced, a frequency shift due to the initial phase of the sinusoids will become apparent.

5 Covariance and Modified Covariance Methods:

The covariance method for AR spectrum estimation is based on minimizing the forward prediction error. The modified covariance method is based on minimizing the forward and backward prediction errors. They are nearly identical, even for a short signal length:

load mtlb
pcov(mtlb(1:64),14,1024,fs)
pmcov(mtlb(1:64),14,1024,fs)

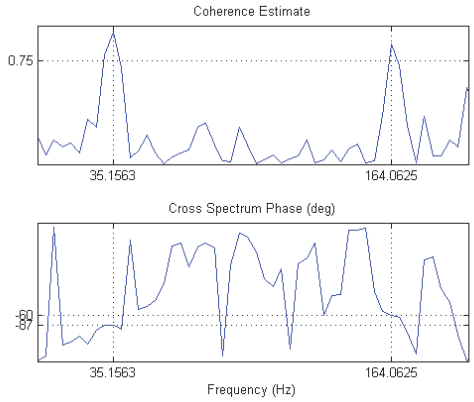


Figure 6. Coherence factor for Estimating Frequency

Consider a number of complex sinusoids embedded in white noise. You can write the autocorrelation matrix *R* for this system as the sum of the signal autocorrelation matrix (*S*) and the noise autocorrelation matrix (*W*):

R = S + W. There is a close relationship between the eigenvectors of the signal autocorrelation matrix and the signal and noise subspaces. The eigenvectors *v* of *S* span the same signal subspace as the signal vectors. If the system contains *M* complex sinusoids and the order of the autocorrelation matrix is *p*, eigenvectors v_{M+1} through v_{p+1} span the noise subspace of the autocorrelation matrix.

5.1 Frequency Estimator Functions:

To generate their frequency estimates, methods calculate functions of the vectors in the signal and noise subspaces. Techniques choose a function that

goes to infinity (denominator goes to zero) at one of the sinusoidal frequencies in the input signal.

$$e(f) = [1 \quad e^{j2\pi f} \quad e^{j4\pi f} \ldots e^{j2(M-1)\pi}]^T$$

The expression $e(f)^H v_k$ is equivalent to a Fourier transform (the vector e(*f*) consists of complex exponentials). The EV method weights the summation by the eigenvalues of the correlation matrix:

$$\hat{P}_{\mathrm{EV}}(f) = \frac{\lambda_k}{\sum_{k=p+1}^{M} |e^H(f) v_k|^2}$$

As metioned above, in addition to estimating the spectrum itself, an estimate of the confidence interval of the spectrum can be generated using a jackknifing procedure.Let us define the following:

Simple sample estimate

$$\hat{\theta} = \frac{1}{n} \sum_i Y_i$$

This is the parameter estimate averaged from all the samples in the distribution (all the tapered spectra).

leave-one-out measurement

$$\hat{\theta}_{-i} = \frac{1}{n-1} \sum_{k \neq i} Y_k$$

This defines a group of estimates, where each sample is based on leaving one measurement (one tapered spectrum) out.Pseudovalues

$$\hat{\theta}_i = n\hat{\theta} - (n-1)\hat{\theta}_{-i}$$

The jackknifed esimator is computed as:

$$\tilde{\theta} = \frac{1}{n} \sum_i \hat{\theta}_i = n\hat{\theta} - \frac{n-1}{n} \sum_i \hat{\theta}_{-i}$$

This estimator is known to be distributed about the true parameter theta approximately as a Student's t distribution, with standard error defined as:

$$s^2 = \frac{n-1}{n} \sum_i \left(\hat{\theta}_i - \tilde{\theta}\right)^2$$

And degrees of freedom which depend on the number of tapers used (Kmax-1):

6 Spectrum calculation

6.1 Spectral analysis using MatLab

You can calculate a power spectrum for x by using the following set of commands:

```
load x552spec
x=x552spec;
P=spectrum(x,m,noverlap);
```

here m is the length of the FFT you want to use-it must be a power of 2 (e.g. 4096,2048,1024,512,256,128,64,32) overlap is the overlap between the blocks of length m-the amount by which adjacent blocks overlap. Since a Hanning window is used, the data near the ends of each block are not weighted very heavily, and it may make sense to have some overlap. You have to take this into account in your statistical testing though.So the two design parameters you have to work with are m and noverlap.

specplot(P,1.) will plot the spectrum.To do cross spectral analysis you can add a second variable name to the command.

P=spectrum(x,y,m,noverlap);

6.2 Seek the Peaks

First look for the periodicities you expect to see in x. First decide what confidence level you require. I suggest either 95% or 99%.

6.3 Cross-Spectral Analysis

Now we want to search for relationships between the two time series x and y. The command to compute the cross spectral analysis is,

```
P=spectrum(x,y,m,noverlap);
specplot(P,1.)
```

you touch the spacebar to run successively through plots of the power spectra of x and y, the amplitude of the cross spectrum, the phase between x and y and the coherence between x and y.You can make a vector of the fre-

quencies in the folowing way,*f=0.5*(0:m2)/m2* (where m2 is m/2, the number of frequencies resolved).Then you would make a linear plot of the first half of the coherency spectrum by typing,

plot(f(1:m4),P(1:m4,5))
where m4=m/4. To plot all resolved frequencies replace m4 with m2
plot(f(1:m2),P(1:m2,5)) to plot the phase, type
ph=angle(P(1:m2,4));
plot(f(1:m4),ph(1:m4))
The phase is given in radians. If you want it in degrees, type
*ph=ph*180/3.141593*
plot(f(1:m4),pd(1:m4))

6.4 Calculation of Spectrum using of Time Series Models:

Syntax
spectrum(sys)
spectrum(sys,{wmin,wmax})
spectrum(sys,w)
spectrum(sys1,...,sysN,w)
ps=spectrum(sys,w)
[ps,w]=spectrum(sys)
[ps,w,sdps] = spectrum(sys)

Description

spectrum(sys) creates an output power spectrum plot of the identified time series model sys. The frequency range and number of points are chosen automatically.sys is a time series model, which represents the system:

$$y(t) = He(t)$$

Where, e(t) is a Gaussian white noise and y(t) is the observed output. spectrum plots abs(H'H), scaled by the variance of e(t) and the sample time.If sys is an input-output model, it represents the system:

$$y(t) = Gu(t) + He(t)$$

Where, u(t) is the measured input, e(t) is a Gaussian white noise and `y(t)` is the observed output.In this case, spectrum plots the spectrum of the disturbance component He(t).spectrum(sys,{wmin, wmax}) creates a spectrum plot for frequencies ranging from wmin to wmax. spectrum(sys,w) creates a spectrum plot using the frequencies specified in the vector w.spectrum(sys1,...,sysN,w) creates a spectrum plot of several identified models on a single plot.

```
%% Time specifications: Fs = 1e4; % samples per second dt = 1/Fs; % seconds per sample StopTime = 1; % seconds t = (0:dt:StopTime-dt)'; N = size(t,1);
  %% Sine wave:
  Fc = 1e3;    % hertz
  x = sin(2*pi*Fc*t);
  noisy_x = x + 0.5*randn(size(x));
  y = xcorr(noisy_x,5e3,'biased');
  display(size(y));
  subplot(2,1,1);
  plot(t,x);
  title('Sine signal');
  subplot(2,1,2);
  plot(t,noisy_x);
  title('Noisy signal');
  %% Fourier Transform:
  X = fftshift(fft(y));
  %% Frequency specifications:
  dF = Fs/N;                    % hertz
  f = -Fs/2:dF:Fs/2;            % hertz
  display(size(f));
  display(size(X));
  %% Plot the spectrum:
  figure;
  plot(f,abs(X)/N);
  xlabel('Frequency (in hertz)');
  title('Autocorrelation spectrum');
```

7 Conclusion

In urban mobile cellular systems, especially when the cell size becomes relatively small, the handoff procedure has a significant impact on system performance. In seamless Mobility handovers there will be no connectivity break but having some additional delay in service, it Degrades Quality of Service. Due to the Prior Time series based Estimation methods of Spectrum. The failures and maximum utilization of Spectrum avoided in seamless Mobility Cellular communication.

References:

1. S. Tekinay and B. Jabbari, *Handover and channel assignment in mobile cellular networks,* IEEE Communications Magazine, vol. 29, pp. 42–46, November 1991.
2. N. D. Tripathi, J. H. Reed, and H. F. VanLandinoham, *Handoff in cellular systems,* IEEE Personal Communications, vol. 5, pp. 26–37, December 1998.
3. R. Vijayan and J. Holtzman, *A model for analyzing handoff algorithms,* IEEE Transactions on Vehicular Technology, vol. 42, pp. 351–356, August 1993.
4. J. Wang, J. Liu, and Y. Cen, *Handoff algorithms in dynamic spreading WCDMA system supporting multimedia traffic,* IEEE Journal on Selected Areas in Communications, vol. 21, pp
5. S. Tekinay and B. Jabbari, *A measurement-based prioritization scheme for handovers in mobile cellular networks,* IEEE Journal on Selected Areas in Communications, 1992.
6. F. Santucci, M. Pratesi, M. Ruggieri, and F. Graziosi, *A general analysis of signal strength handover algorithms with cochannel interference,* IEEE

DE GRUYTER
OPEN

JACSM 2014, Vol. 6, No. 1, pp. 27 - 42
DOI 10.2478/jacsm-2014-0002

PERSISTENT COLLECTIONS WITH CUSTOMIZABLE EQUIVALENCE AND IDENTITY SEMANTICS

Konrad Grzanek

IT Institute, University of Social Sciences, Łódź, Poland
kgrzanek@spoleczna.pl, kongra@gmail.com

Abstract

Providing a comprehensive set of mechanisms solving the problem of controlling equivalence and identity requires implementing the functionality for non-sequential containers instrumented with the enriched semantics. Functional programming languages, like Clojure, typically miss the functionality by default. The article presents the design considerations, concepts and implementation details of generalized sets and maps aware of the customizable equivalence and identity together with some usage examples.

Key words: Equivalence testing, semantics, identity, functional programming, Clojure, persistent collections

1 Introduction

Testing object for their equivalence as well as creating criteria of establishing their identity is one of the most important tasks to be realized during mature software implementation process. A developer must consider many factors here, like the typing system of the programming language, either static or dynamic, based on the structural equivalence or using tags. The number and the extent of decisions to be made when designing the identity and equivalence related algorithms is so large and wide that the attempts to make a kind of their uniformization in a single library or *API* requires special approaches. Namely, the mechanisms must be put in a formal shape consisting of a set of extendable equivalence and identity operators. A lack of these mechanisms even in as mature programming languages as *Java* and *Clojure* led to a work on their detailed design and implementation in the latter.

The results of the undertaking were presented in a paper titled *Equivalence in Java and Clojure, Design and Implementation Considerations* [1]. Most of it's content focused on:

- the equivalence and identity abstractions,
- implementation details for Java *primitive types* [3],
- implementation details for *java.lang.Number* [2] derivatives,
- implementation details for some *Clojure* [4], [5] reference types (*clojure.lang.Ratio*, *clojure.lang.BigInt*),
- implementation details for selected *sequential values*: instances of *java.lang.String*, *clojure.lang.ISeq*, *java.util.List*, *kongra.core.Pair*.

The mentioned paper lacked a presentation of concepts as well as implementation elements for other kinds of collections, namely *sets* and *maps* (*associative containers*). The present article may be treated thereafter as a natural continuation of [1] with an aim to present customizable, extendable equivalence and identity realization in the two kinds of containers. One final mark to be made here is we will focus on Clojure persistent *sets* and *maps*, abandoning completely other realizations of *java.util.Set* and *java.util.Map* interfaces [2].

2 Equivalence, Identity and the Persistent Collections in Clojure

Traditionally the functional programming style is associated with a lack of explicitly changeable state. Objects posses *value semantics*, they are immutable *values* like the notions in mathematics. This property opens ways to discuss formally the properties of programs and even sometimes makes the properties of programs provable. From the engineer's point of view the lack of state makes achieving programs' correctness easier (in more complex cases – possible).

Immutable collections are the core of Clojure data structures [5]. They are called *persistent* there, with the term *persistent* meaning their persistence across the operators working on them. In other words, the collection operators in the language are *non-destructive*, they do not modify their arguments. Another property of the persistent collections is their *structure-sharing*. This is a pretty old concept, sequences with structure sharing were common in various dialects of Lisp (see [7], [8]). Nowadays data structures other than the sequential ones are known to have the property, e.g. *hash-tries* [6]. Clojure has the following kinds of persistent collections:

– sequences, including vectors1
– sets
– maps
– records – tagged objects with a map semantics

As stated earlier, our previous paper [1] presented a realization of customizable equivalence and identity operators for sequences (*clojure.lang.ISeq* derivatives) and lists including vectors (*clojure.lang.PersistentVector* <: *clojure.lang.APersistentVector* <: *java.util.List*). As a consequence of having these mechanisms:

```
> (deep= [1M 2N] [1 2])
true
> (deep-hash [1M 2N])
994
> (deep-hash [1 2])
994
```

while with the default operators we have erroneous:

```
> (= [1M 2N] [1 2])
false
> (hash [1M 2N])
-1806861044
> (hash [1 2])
156247261
```

Unfortunately, with persistent sets and maps, things go wrong:

```
(deep= #{1M 2N} #{1 2})
```

```
No implementation of method: :binary-deep= of protocol:
#'kongra.identity/WithDeep= found for class: clojure.lang.PersistentHashSet
  [Thrown class java.lang.IllegalArgumentException]
```
[2]

and similarly:

```
(deep= {1M 2N} {1 2})
No implementation of method: :binary-deep= of protocol:
#'kongra.identity/WithDeep= found for class: clojure.lang.PersistentArrayMap
  [Thrown class java.lang.IllegalArgumentException]
```
[2]

1 In an opposition to other sequences, vector elements may be randomly accessed in constant time using indexes, yet the vectors are not lazily evaluated, like other types of sequences may be.

2 Observable when being run either in a standalone application or in the Clojure *REPL*

The same lack of implementation occurs for *kongra.identity/deep-hash* operator. As expected, the default Clojure equivalence and identity operators, exhibit wrongful behavior:

```
> (= #{1M 2N} #{1 2})
false
> (hash #{1M 2N})
1271160563
> (hash #{1 2})
460223544
```

and

```
> (= {1M 2N} {1 2})
false
> (hash {1M 2N})
820711855
> (hash {1 2})
1952097988
```

A thoughtful reader might ask was this lack of implementation an accident or a deliberate decision. The "deep" operators throw errors after all when called with arguments of unsupported kinds. In fact it was a deliberate decision dictated by the following two considerations:

1. There should be no default and "safe" implementation for objects of arbitrary types. Raising errors allows to find out and eliminate unpredictability in the system eagerly.
2. It is impossible to provide the implementation of the customizable equivalence and identity operators for arbitrary *set* or *map* type.

The second point is particularly interesting. In the case of *sequences* and *lists* (see [1]) the way to achieve the desired functionality was iterating over the elements of the collection and either aggregating their *deep-hash* values into a resulting *deep-hash* for the sequence or testing for the *deep=* (equivalence) of the (deeply) compared sequences. This approach must be extrapolated onto sets and maps as it seems to be the only reasonable implementation strategy; a *deep-hash* or *deep=* for any collection is a derivative of *deep-hash* or *deep=* of it's components. Unfortunately, due to various ways a set or a map may be implemented (*hash codes*, *balanced trees*), there is no easy way

to provide the “deep” semantics into these kinds of collections[3]; one can't inject neither *deep-hash* nor *deep=* into their internal workings[4].

All this led to making another undertaking of implementing special flavors of persistent sets and maps characterized by:

- possessing the *deep-hash/deep=* semantics,
- generosity in their implementation,
- ability to derive the actual container implementation by relying transparently on a specific *back-end* collection.

The following sections constitute a detailed description of the design decisions and a presentation of their implementation details.

3 DeepEntry

The major idea behind the realization of our sets and maps was to create wrapper instances that would be the actual representations of the objects to store within the container. Using this approach we could forget about reaching for the storage internals at the same time not loosing the ability to inject the customizable equivalence and identity semantics. We may call the wrapper instances the *proxy* objects (behind the *proxy* design pattern as described in [9]), because they allow to modify the original behavior of the stored objects.

The proxy type is called *DeepEntry*. The instances of this class are immutable POJO (*Plain-Old Java Objects*). The following listing shows the static structure of the class:

```
public class DeepEntry {

  private final Object value;

  private int hash; // Default to 0

  private DeepEntry(Object value) {
    this.value = value;
  }
}
```

3 In an opposition to a simple sequence or list a set or map exhibits a whole variety of “views” of it's data in a form of iterators, entry sets, etc. It is a natural and justifiable expectation to these derivative “views” to also provide the “deep” semantics.

4 Testing a set or a map for containing a specified element is the most problematic functionality, as the test is associated with using *deep-hash* and/or *deep=* instead of default *java.lang.Object* methods *hashCode* and/or *equals* in the core storage for these data structures.

Deep entries simply store the value and they offer caching of hash codes, like other immutable Java classes, e.g. *java.lang.String*. The two overridden methods are the core of the implementation:

```
  @Override
  public int hashCode() {
    if (0 == hash && null != value) {
      hash = (Integer) DeepHash.proxy.invoke(value);
    }
    return hash;
  }

  @Override
  public boolean equals(Object obj) {
    if (this == obj) {
      return true;
    }
    final Object result = DeepEquiv.proxy.invoke(
                    value, DeepEntry.uncast(obj));
    return RT.booleanCast(result);
  }
```

As it can be seen, both the identity as well as the equivalence operators expressed this way use the *deep-hash* and *deep=* procedures, represented here by special Java constants *DeepEquiv.proxy* and *DeepHash.proxy*.

DeepEntry belongs in fact to the implementation internals, it has a private constructor, but still there are the useful operators that allow any object to be *coerced* to a *DeepEntry*:

```
public static DeepEntry cast(Object obj) {
  if (obj != null &&
      obj.getClass() == DeepEntry.class) {
    return (DeepEntry) obj;
  }
  return new DeepEntry(obj);
}
```

and *uncasted* afterward if needed:

```
public static Object uncast(Object obj) {
  if (obj != null &&
      obj.getClass() == DeepEntry.class) {
    return ((DeepEntry) obj).value;
  }
  return obj;
}
```

The latter mechanism is also extended to a form of a Clojure procedure, as this procedure is a part of the implementation (referenced to in the following sections):

```
public static final AFn uncastFn = new AFn() {
  @Override
  public Object invoke(Object arg) {
    return uncast(arg);
  }
};
```

4 DeepSet

A *DeepSet*, the set that uses the customizable equivalence and identity semantics is the first kind of collection to be described. In fact it's internals are relatively simple to present.

The construction of the class mimics the implementations of other *persistent sets* within the standard Clojure library – the class is effectively *non-extendable* (by the presence of only private constructor), it has the *value semantics* and it implements a set of useful interfaces:

```
public class DeepSet extends AFn implements IObj,
             IPersistentSet, Set, IHashEq {

  private final APersistentSet impl;

  private DeepSet(APersistentSet impl) {
    this.impl = impl;
  }

}
```

As it can be seen above, a *DeepSet* instance wraps a persistent set (the class *clojure.lang.APersistentSet* is the persistent set abstraction) called ***impl*** here. All objects stored in the *DeepSet* are being actually put into the persistent set after being coerced to a *DeepEntry*. This is a design pattern common both for *DeepSets* and for their associative counterpart – the *DeepMap* (see the next section). A manifestation of this approach can be seen at the following listing presenting the process of creating the set:

```
public static DeepSet create(APersistentSet impl) {
  return new DeepSet(impl);
}
```

```
public static DeepSet create(APersistentSet impl,
                             Object... elements) {
  if (elements.length == 0) {
   return new DeepSet(impl);
  }

  for (Object object : elements) {
   impl = (APersistentSet) impl.cons(
                           DeepEntry.cast(object));
  }
  return new DeepSet(impl);
}
```

The underlined part of the code is the process of performing the actual storage of an object with simultaneous coercing to *DeepEntry*.

With the presence of this approach, implementing equivalence operators for *DeepSet* is trivial; it boils down to calling proper equivalence operators on the underlying ***impls***.

```
@Override
public boolean equiv(Object obj) {
  if (this == obj) {
    return true;
  }
  if (!(obj instanceof DeepSet)) {
    return false;
  }
  DeepSet other = (DeepSet) obj;
  return this.impl.equals(other.impl);
}

@Override
public boolean equals(Object obj) {
  return this.equiv(obj);
}
```

The same can be told about computing the *hash code* using the customizable mechanisms – the work is done in the ***impl*** with the presence of *DeepEntries* and their semantics:

```
@Override
public int hashCode() {
  return impl.hashCode();
}
```

```
@Override
public int hasheq() {
  return hashCode();
}
```

On the Clojure side there is a collection of mechanisms for creating DeepSet instances:

```
(defn deep-hash-set
  ([]
     (DeepSet/create (hash-set)))

  ([& keys]
     (DeepSet/create ^APersistentSet (hash-set) keys)))

  and

(defn deep-sorted-set
  [& keys]
  (DeepSet/create ^APersistentSet
        (sorted-set-by (deep-comparator))
        keys))

(defn deep-sorted-set-by
  [pred & keys]
  ((DeepSet/create ^APersistentSet
        (sorted-set-by (deep-comparator pred))
        keys)))
```

The latter two implemented with the help of a special constructor of *DeepEntry* aware comparators:

```
(defn deep-comparator
  ([]
     (deep-comparator compare))

  ([pred]
     (fn [e1 e2]
       (pred (DeepEntry/uncast e1)
             (DeepEntry/uncast e2)))))
```

There are also the procedures for casting collections into *DeepSets*:

```
(defn deep-set?
  [x]
  (instance? DeepSet x))
```

```
(defn ^DeepSet deep-set
  [coll]
  (if (deep-set? coll)
    coll

    (apply deep-hash-set (seq coll))))
```

and a predefined empty *DeepSet*:

```
(def EMPTY-DEEP-SET (deep-hash-set))
```

All the destructive *java.lang.Set* operators are prohibited on *DeepSet*, as it is a persistent, immutable type. This is why:

```
@Override
public boolean add(Object e) {
  throw new UnsupportedOperationException();
}

@Override
public boolean remove(Object o) {
  throw new UnsupportedOperationException();
}

@Override
public boolean addAll(Collection c) {
  throw new UnsupportedOperationException();
}

@Override
public boolean removeAll(Collection c) {
  throw new UnsupportedOperationException();
}

@Override
public boolean retainAll(Collection c) {
  throw new UnsupportedOperationException();
}

@Override
public void clear() {
  throw new UnsupportedOperationException();
}
```

Finally, after extending the proper protocols[5], namely *WithDeepHash* and *WithDeep=* (see [1]), we get:

```
> (in-ns 'kongra.identity)

> (def s1 (deep-hash-set 1M 2N))
> s1
#{1M 2N}

> (def s2 (deep-hash-set 1 2))
> s2
#{1 2}

> (deep-hash s1)
3
> (deep-hash s2)
3
> (deep= s1 s2)
true
```

Additionally, the semantics expands onto the standard Clojure identity and equivalence operators:

```
> (= s1 s2)
true
> (hash s1)
3
> (hash s2)
3
```

5 DeepMap

The implementation of an associative container differs from the *DeepSet* mostly in the fact that here we store both keys and values, both wrapped within the *DeepEntry*. The *DeepMap* also uses ***impl*** – an internal *back-end* storage, but here it is a persistent map (*clojure.lang.APersistentMap* derivative):

```
public class DeepMap extends AFn implements
         IObj, IPersistentMap, Map,
         Serializable, MapEquivalence, IHashEq {

  private final APersistentMap impl;
```

5 Tedious to present here due to a large amount of source code. For more, see the *kongra/identity.clj* compilation unit.

```
  private DeepMap(APersistentMap impl) {
    this.impl = impl;
  }
}
```

The identity (hash) and equivalence operators are trivial, as in the case of the *DeepSet*:

```
@Override
public boolean equiv(Object obj) {
  if (this == obj) {
    return true;
  }
  if (!(obj instanceof DeepMap)) {
    return false;
  }
  DeepMap other = (DeepMap) obj;
  return this.impl.equals(other.impl);
}

@Override
public boolean equals(Object obj) {
  return this.equiv(obj);
}

@Override
public int hashCode() {
  return impl.hashCode();
}

@Override
public int hasheq() {
  return hashCode();
}
```

Creating *DeepMaps* is associated with performing the process of putting the keys and values into the ***impl*** after wrapping them. One can observe this at the following piece of code:

```
public static DeepMap create(APersistentMap impl,
                             Map other) {
  if (null == other) {
    return create(impl);
  }

  for (Object obj : other.entrySet()) {
    Map.Entry entry = (Map.Entry) obj;
```

```
    impl =
        (APersistentMap) impl.assoc(
          DeepEntry.cast(entry.getKey()),
          DeepEntry.cast(entry.getValue()));
  }
  return create(impl);
}
```

For the associative containers we also have a collection of operators on the Clojure side, that allow to create the “deep” maps:

```
(defn deep-hash-map
  ([] (DeepMap/create (hash-map)))

  ([& keyvals]
     (DeepMap/create ^APersistentMap (hash-map)
                     keyvals)))

(defn deep-sorted-map
  [& keyvals]
  (DeepMap/create ^APersistentMap (sorted-map-by
                                    (deep-comparator))
                  keyvals))

(defn deep-sorted-map-by
  [pred & keyvals]
  (DeepMap/create ^APersistentMap (sorted-map-by
                       (deep-comparator pred))
                   keyvals))
```

or casting arbitrary maps into their “deep” counterparts:

```
(defn ^DeepMap deep-map
  [m]
  (cond (deep-map? m)
        m

        (instance? clojure.lang.Sorted m)
        (let [c (.comparator ^clojure.lang.Sorted m)]
          (DeepMap/create ^APersistentMap
                      (sorted-map-by (deep-comparator c))
                      ^java.util.Map m))

        (instance? java.util.SortedMap m)
        (let [c (.comparator ^java.util.SortedMap m)]
          (DeepMap/create ^APersistentMap
                      (sorted-map-by (deep-comparator c))
```

```
                    ^java.util.Map m))
        :else
        (DeepMap/create ^APersistentMap (hash-map)
                        ^java.util.Map m)))
```

As in the case of the *DeepSet*, *DeepMaps* ensure their non-destructive nature by raising exceptions on an attempt to call destructive operators (belonging to the *java.util.Map* contract). For the sake of simplicity we do not present the related source codes here.

Testing for keys and values containment is as trivial as in the case of the identity and equivalence implementation after wrapping the arguments withind *DeepEntry*:

```
@Override
public boolean containsKey(Object key) {
  return impl.containsKey(DeepEntry.cast(key));
}

@Override
public boolean containsValue(Object value) {
  return impl.containsValue(DeepEntry.cast(value));
}
```

and similarly in the case of retrieving values stored under the given keys:

```
@Override
public Object get(Object key) {
  return DeepEntry.uncast(impl.get(DeepEntry.cast(key)));
}
```

One final interesting implementation aspect is the algorithm for building the derivative collections of *map* entries, keys and values. They are all built around the same pattern – instances of *kongra.utils.decorators.ImmutableDecoratingSet*[6] are used as shown below:

```
@SuppressWarnings("unchecked")
@Override
public Set entrySet() {
  return new ImmutableDecoratingSet<Map.Entry,
                                    Map.Entry>
              (impl.entrySet()) {
    @Override
    protected Map.Entry decorate(Object o) {
      Map.Entry entry = (Map.Entry) o;
```

6 Full presentation of this interesting collection, whose range of possible applications is by no means limited to the functionality presented here, goes beyond the scope of this article.

```
      return new MapEntry(DeepEntry.uncast(
                              entry.getKey()),
          DeepEntry.uncast(entry.getValue()));
    }
  };
}
@SuppressWarnings("unchecked")
@Override
public Set keySet() {
  return new ImmutableDecoratingSet(impl.keySet()) {
    @Override
    protected Object decorate(Object o) {
      return DeepEntry.uncast(o);
    }
  };
}
```

or – in the case of *java.util.Map.values* method implementation – the *kongra.utils.decorators.ImmutableDecoratingCollection*[6] is used instead:

```
@SuppressWarnings("unchecked")
@Override
public Collection values() {
  return new ImmutableDecoratingCollection(
              impl.values()) {
    @Override
    protected Object decorate(Object o) {
      return DeepEntry.uncast(o);
    }
  };
}
```

There are few more elements of the implementation of the *DeepMap* container whose presentation in this paper was postponed for the sake of the overall clarity.

Using the container is straightforward and it leads to the desired behaviors:

```
> (def m1 (deep-hash-map 1M 2N))
> m1
{1M 2N}
> (def m2 (deep-hash-map 1 2))
> m2
{1 2}
> (deep= m1 m2)
true
> (deep-hash m1)
3
```

```
> (deep-hash m2)
3
```

As with *DeepSets*, here we also have:

```
> (= m1 m2)
true
> (hash m1)
3
> (hash m2)
3
```

6 Conclusions

This paper finalizes a series of articles related to the customizable equivalence and identity in Clojure. The presented mechanisms should be used where needed, optionally extended on demand by modifying and/or extending the protocols presented in [1] and referenced to in this article. Establishing the specific performance profiles of the algorithms shown is left to the discretion of their users.

References

1. Grzanek K., 2013, *Identity in Java and Clojure, Design and Implementation Considerations*, Journal of Applied Computer Science Methods, No. 2 Vol. 5 2013
2. Oracle, 2014, *Java™ Platform, Standard Edition 8 API Specification*, http://docs.oracle.com/javase/8/docs/api/
3. Gosling J., Joy B., Steele G., Bracha G., 2005, *The Java™ Language Specification Third Edition*, ISBN 0-321-24678-0, available at the Oracle Technology Network (2014) http://docs.oracle.com/javase/specs/
4. Halloway S., 2009, *Programming Clojure*, ISBN: 978-1-93435-633-3, The Pragmatic Bookshelf
5. Emerick Ch., Carper B., Grand Ch., 2012, *Clojure Programming*, O'Reilly Media Inc., ISBN: 978-1-449-39470-7
6. Bagwell P., 2000, *Ideal Hash Trees (Report)*, Infoscience Department, École Polytechnique Fédérale de Lausanne
7. Touretzky D.S., 1990, *COMMON LISP: A Gentle Introduction to Symbolic Computation*, The Benjamin/Cummings Publishing Company,Inc., ISBN: 0-8053-0492-4
8. Graham P., 1993, *On Lisp - Advanced Techniques for Common Lisp*, Prentice Hall; 1st edition (September 9, 1993), ISBN-10: 0130305529, ISBN-13: 978-0130305527
9. Gamma, et al., E., 1995., *Design Patterns*. Reading, MA: Addison-Wesley Publishing Co, Inc. pp. 175ff. ISBN: 0-201-63361-2

DE GRUYTER
OPEN

JACSM 2014, Vol. 6, No. 1, pp. 43 - 53
DOI 10.2478/jacsm-2014-0003

THE APPLICATION OF THE LAPLACE TRANSFORM FOR MODELING OF GAS FLOW USING MAPLE®

Małgorzata Wójcik[1], Mirosław Szukiewicz[1], Paweł Kowalik[2]

[1] Department of Chemical and Process Engineering
Rzeszow University of Technology, Rzeszow, Poland
wojcik.m@aol.com,ichms@prz.edu.pl

[2] Institute of New Chemical Synthesis, Pulawy, Poland
pawel.kowalik@ins.pulawy.pl

Abstract

The Laplace transform is powerful method for solving differential equations. This paper presents the application of Laplace transform to solve the mathematical model of gas flow through the measuring system. The basis of the mathematical model used to describe and simulate the analyzed process is a system of ordinary differential equations. As a result is obtained a single algebraic equation in terms of the complex variable *s*. In order to study the dynamics of the system these equation should be reverted to the time domain by performing an inverse Laplace transform. Calculations were performed in the environment of Maple®. Determining of mixing in a u-shape vessel is aim of presented calculations.

Key words: Laplace transform, mathematical model, differential equations, Maple program

1 Introduction

The Laplace transform is an important integral transform with many applications in mathematics, physics, engineering etc. The Laplace transform is powerful tool of solving computational problems. For example, this method can be used for solution and analysis of time – invariant systems such as electrical circuits, mechanical systems, optical devices and harmonic oscillators. Primarily, this transform is very attractive in solving differential equations and therefore play important role in automatics and control theory.

Mathematical modeling is an important theoretical approach in studying problems. Generally, it involves finding the solutions to the mathematical model constructed to investigate the problem of interest. Usually, a mathematical model consists of a set of differential equations mathematically describing the physical conditions of the problem and a set of boundary and initial conditions appropriately prescribed. Many analytical solutions for various mathematical models of chemical process have been obtained using the Laplace transform technique.

The Laplace transform has been studied by many authors over the years due to the wide array of applications of the Laplace transform technique to different areas of science. There have now been many publications, for example the book written by Widder (1941) [1] about the Laplace transform and inversion formulas, the book written by Jaeger in 1961 [2] describes applied the Laplace transform with engineering applications. A nice review of this transform applied to Ordinary Differential Equations (ODEs) and Partial Differential Equations (PDEs) is given in Poularikas [3]. There are also articles describing the use of the Laplace transform in chemical engineering, e.g. articles written by Kolev and Linden [4], Ahmed and Batin [5], Ahmed and Kalita [6], Membrez and others [7]. Kolev and Linden [4] used Laplace transform for the solution of partial equations describing the transient mass-transfer in laminar flow systems and heat-transfer in single and multi-stream flow systems. The papers [5] and [6] describe the use of the Laplace transform for analysis the transient convection-radiation magnetohydrodynamic viscous flow in a porous medium and study hydromagnetic flow in chemical reactors. Membrez and others [7] used the Laplace transform technique of find the kinetic parameters for the adsorption of a protein on porous beads. A bibliography of a great many papers are available on the WEB.

In this paper, application of Laplace transform technique for analysis of the process of two gases mixing on the basis of signal given by TCD-type detector is presented. The actual investigation have two main aims. The first one is determination of gas mixing in the continuous flow vessel. And the second one is checking out the hypothesis that the Laplace transform makes easier solution finding and the analysis of the answer of the analytic system.

2 Definition and notation of the Laplace transform

The Laplace transform of a function f(t) is formally defined as

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}dt \qquad (1)$$

where *s* is a complex variable corresponding to time. In the above formula (1), the function e^{-st} is defined as the kernel of the integral of Laplace. The integral $\int_0^\infty f(t)e^{-st}dt$ is called the Laplace integral of the function *f(t)*. The symbol $\mathcal{L}$ is the Laplace transformation, which generates a new function *F(s)* = $\mathcal{L}\{f(t)\}$.

Applications of the Laplace transform to the mathematical models simplifies the models by reducing the degree of freedom of the independent variable. The various types of problems that can be treated with the Laplace transform include ordinary and partial differential equations as well as integral equations. A differential equation of the model for a physical system can be subjected to the Laplace transform in order to produce an algebraic form of model in the transform variable *s*. The solutions for the transformed models in the Laplace domain can be easier obtained. The most difficult step is inverse Laplace transform finding. The Laplace transform method requires usually much less work than classical methods. Computer Algebra Systems (CAS) programs usually operate the method.

3 Model

Presented here investigations are the part of larger research task and it seems to be purposeful to present main problem of the task. Presented in Figure 1 research unit was developed to determine a value of effective diffusion coefficient in a catalyst pellet. The idea of the experiment is new, it has not been described in literature. The procedure of main experiment will be the same as described below, in the point 3.2 with the only difference namely the column will be filled with catalyst pellets. Under experimental conditions two crucial factors will determine a shape of recording signal. They are: (i) effective diffusion coefficient in the catalyst pellets and (ii) mixing in the zones 1, 3 and 4. The proper evaluation of effective diffusion coefficient needs separating of influence of mixing. To determine it were made investigations described below, carrying on without catalyst. It was the main goal of the described experiments. The second aim was finding easy and effective method of the model analysis and solution.

The influence of mixing in the unit can be determined in the following way. Each zone is divided into n cells, a fluent is perfectly mixed in each cell. Cells are placed in series. The more cells in a zone, the flow better approaches plug flow. Matching the model solution for chosen number of the cells with curves obtained as a results of experiments one can determine mixing in the unit. Presented way is very convenient, moreover the number of cells the best fitting experimental results can take into consideration also parameters or factors with no reflection in model equations.

3.1 Measuring system

Scheme of the measuring system is presented in Figure 1.

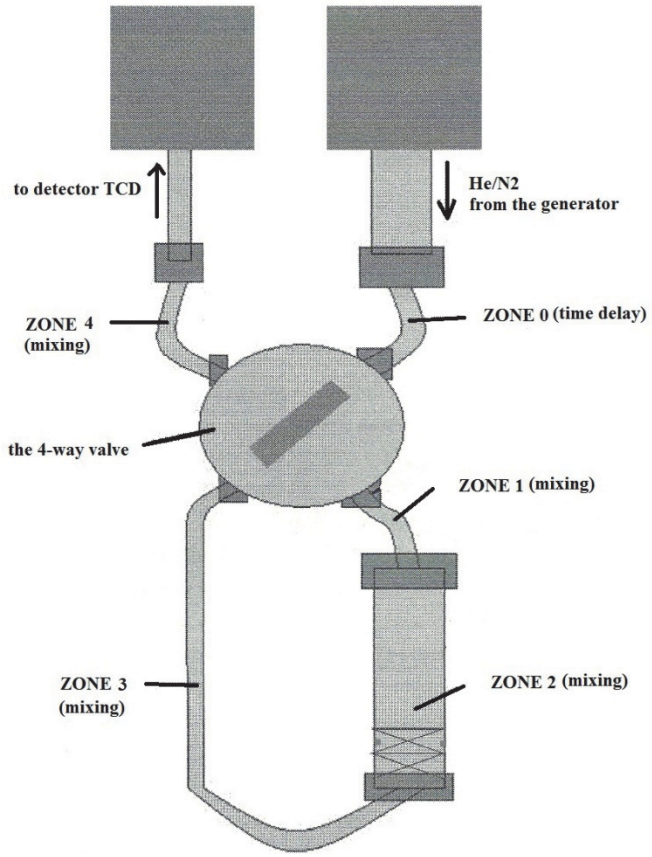


Figure 1. The scheme of the measuring system with u-shape vessel.

The unit consists of the following elements, the 4-way valve, u-shape vessel (empty in the actual investigations), thermal conductivity detector (TCD) and pipes connected the mentioned elements. The system was divided onto five zones; they are distinguished on basis of geometry and/or its the function.

Zone 0: the pipe connected inlet of gas and the 4-way valve;
the length of the zone: 0.75dm, the diameter of the zone: 0.0125dm
Zone 1: the pipe connected the 4-way valve and a column inlet;
the length of the zone: 1.8 dm, the diameter of the zone: 0.0125 dm

Zone 2: empty vessel;
the length of the zone: 1.0 dm, the diameter of the zone: 0.056 dm
Zone 3: the pipe connected a column outlet and the 4-way valve;
the length of the zone: 4.0 dm, the diameter of the zone: 0.0125dm
Zone 4: the pipe connected the 4-way valve and TCD detector;
the length of the zone: 0.75dm, the diameter of the zone: 0.0125dm.

In order to identify the type of mixing, each zone were divided into *n* cells. Number of cells in each zone may be different. One cell in the zone corresponds to the ideal mixing in this zone. Infinite number of cells in the zone corresponds to the plug flow.

3.2 Description of the experiments

The study was conducted as follows. The system was flushed for 10 minutes with a constant flow of helium (flow rate of 0.04 dm^3/min). Then the valve was closed the 4-way valve leading to shut off the flow of the gas through the vessel (Zone 1, 2, 3). For the next 15 minutes the system (beside the vessel) was purged with a nitrogen (flow rate of 0.01dm^3/min) until a stable base TCD signal was reached. After about seven minutes of stabilizing the system again turned over the 4-way valve to allow a constant flow of nitrogen with the volumetric flow rate of 0.01dm^3/min or 0.04 dm^3/min through the vessel. In result ' trapped ' helium was removed and TCD signal was generated and recorded.

3.3 Assumptions of the model

Model describing the process is based on the following assumptions:
- The system is operated under isothermal conditions at constant pressure.
- Gases satisfy the equation of state of an ideal gas.

3.4 Mass balance of the process

Mass balance of the nitrogen in the individual zones and for the number of cells of *n* leads to the following equations:

Zone 0: $c_0(t) = c_{in}(t - t_d)$ (2)

Zone 1: $V_{c1}\frac{dc_{1,1}}{dt} = q(c_0 - c_{1,1}); V_{c1}\frac{dc_{1,k}}{dt} = q(c_{1,k-1} - c_{1,k})$; k=2..n1 (3)

Zone 2: $V_{c2}\frac{dc_{2,1}}{dt} = q(c_{1,n_1} - c_{2,1})$; $V_{c2}\frac{dc_{2,k}}{dt} = q(c_{2,k-1} - c_{2,k})$; k=2..n2 (4)

Zone 3: $V_{c3}\frac{dc_{3,1}}{dt} = q(c_{2,n_2} - c_{3,1})$; $V_{c3}\frac{dc_{3,k}}{dt} = q(c_{3,k-1} - c_{3,k})$; k=2..n3 (5)

Zone 4: $V_{c4}\frac{dc_{4,1}}{dt} = q(c_{3,n_3} - c_{4,1})$; $V_{c4}\frac{dc_{4,k}}{dt} = q(c_{4,k-1} - c_{4,k})$; k=2..n4 (6)

The He/N_2 mixture outlet concentration from the previous cell is the inlet concentration for the next (excluding the first and the last cell in the system). Concentration $c_{4,n4}$ is outlet concentration and is measured by TCD detector. n1, n2, n3 n4 are the numbers of cells in each of zones.
The number of cells in the zones determine the number of equations. Transforming the system of equations into a one equation is difficult and even is not impossible.

For each zone were formulated accordingly initial and boundary conditions:

Zone 1: $c_{1,1}(0) = c_T$; k=1..n1; (7)

Zone 2: $c_{2,1}(0) = c_T$; k=1..n2; (8)

Zone 3: $c_{3,1}(0) = c_T$; k=1..n3; (9)

Zone 4: $c_{4,1}(0) = 0$; k=1..n4; (10)

where:
q - gas flow [dm^3/min]
c_{in} - gas concentration in inlet to zone 0 [mol/dm^3]
V_{ck} - volume of a single cell in *k*-th cell [dm^3]
$c_{j,k}$ - gas concentration in *j*-th zone, in cell *k* [mol/dm^3]
t – time [min]
$c_T = \frac{P}{R_g \cdot T \cdot 10^3}$ = 0.03906 [mol/dm^3]
P – pressure [Pa]
R_g – gas constant [$J \cdot mol^{-1} \cdot K^{-1}$]
T – temperature [K]

And finally inlet concentration is described by

$$c_{in} = \begin{cases} 0 \text{ for } t < 0 \\ c_T \text{ for } t \geq 0 \end{cases} \quad (11)$$

and time delay by

$$t_d = \frac{V_0}{q} \tag{12}$$

where:
V_0 is the volume of zone 0.

3.5 Analysis of the model and its solution

Presented in previous section model is simple, but obtaining a solution can be a little difficult. The variable of interest is $c_{4,n4}$ as measurable concentration. For different values of n1, n2, n3 and n4 the model consist various number of equations. Moreover the values of n1 – n4 have to be determined using trial and error method to obtain the best fit between model solution and experiments. For this reason it is necessary to solve the system repeatedly, the system can contain large number of equations, and finally the number of equations changes for each try. It is very inconvenient situation. One can try to eliminate variables out of interest from the system, but as a result high order differential equation will be obtained and the order of equation will changes for each try. Due to mentioned reasons we paid our attention on well-known tool of analysis and solution of non-stationary models namely Laplace transform. Model transformed into algebraic equations model one can easily solve with respect to variable $c_{4,n4}$, see equation (13)

$$c_{4,n4} = \frac{1}{s} \cdot \left[\left(\frac{q}{n4 \cdot V_{c4} \cdot s + q} \right)^{n4} \left(\frac{q}{n3 \cdot V_{c3} \cdot s + q} \right)^{n3} \left(\frac{q}{n2 \cdot V_{c2} \cdot s + q} \right)^{n2} \left(\frac{q}{n1 \cdot V_{c1} \cdot s + q} \right)^{n1} \cdot c_{in} + \left(\frac{q}{n4 \cdot V_{c4} \cdot s + q} \right)^{n4} \cdot c_{in} \right] \cdot e^{-t_d s} \tag{13}$$

It is noteworthy, that initial conditions and time delay are considered in the equation (13),
where:
s – complex variable (Laplace variable).

It is expected, that Eq. (13) for real conditions will be of high order. In that case a solution will contain many terms and its obtaining would arduous. But currently there exists computer programs, usually called Computer Algebra Systems which can help to obtain model solution. We are use one of this type program, namely Maple®. Maple® operates Laplace transform. Finding the general, valid for any values of n1 .. n4 coefficients, solution was impossible using Maple®. Despite this fact the program was very helpful. Trial and error method calculations were made fast and evaluation of n1..n4 coefficients and thereby evaluation of the mixing did

not cause problems. In Figure 2 is presented screenshot with a single calculation by Maple (data introduction are omitted). It is worth paying attention on scary amount of the lines of Maple code. Presented lines concern model equation, its inversion, error calculation and visualization of results. Data input and "answers" of the program was omitted, excluding visualization of the solution. Average time spent for calculation is equal to 0.06s.

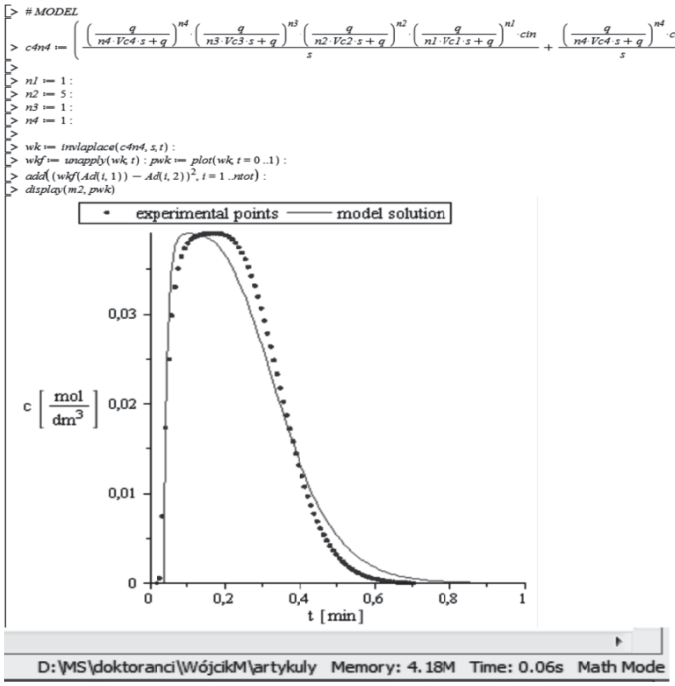


Figure 2. Screenshot of program Maple.

4 Results

On the basis of presented model the following results were obtained. The best fit the smaller gas flow is presented on Figure 3 and for the larger gas flow in Figure 4. the model solution is presented by red solid line while the blue points show the results of experiment. The best fit was determined on the basis of minimal value of sum of squares of differences between calculations and experiments.

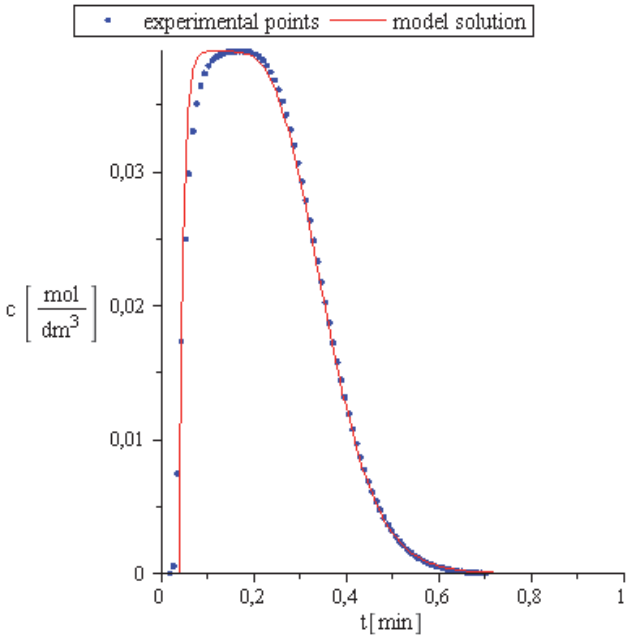


Figure 3. Experimental and theoretical profiles gas concentrations for 0.01dm^3/min and n1=1, n2=12, n3=1, n4=1.

In both cases the obtained fits are very good, differences are small. The best fit for smaller gas flow was obtained for smaller number of cells in the zone 2. This result agrees with the flow theory – the higher velocity of gas the more similar to the plug flow is a real flow. Numbers of cells in zones 1, 3 and 4 are the same and equal to 1. This means that gas in the pipes is perfectly mixed.

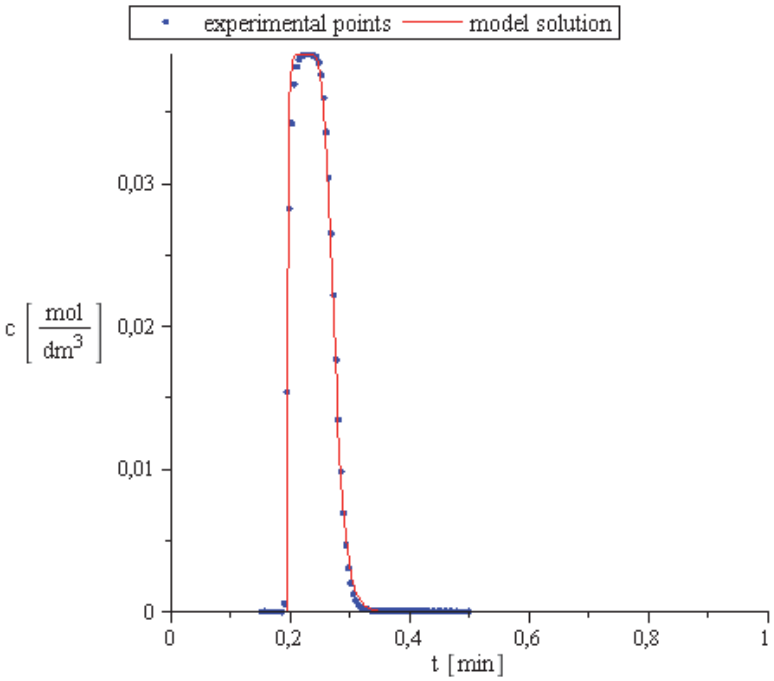


Figure 4. Experimental and theoretical profiles gas concentrations for 0.04dm^3/min and n1=1, n2=40, n3=1, n4=1.

In both cases one can observe similar differences between theoretical and experimental curves. Sloping down part of experimental curve is much better fitted than their sloping up part. In theory sloping up part of the curve is almost vertical in contrast to experimental, especially for the smaller gas flow. It results from the simplicity of the used model which do not consider all theoretically predicted phenomena. In the unit occurs dispersion, phenomenon causing migration of compound from high to low concentration region (analogously to diffusion). As a result of dispersion a slope of a recorded curve is not as sharp as theoretical one. In our opinion precision of the presented model is satisfactory, especially for the higher gas flow. Consideration of dispersion phenomenon in model results in much complicated system of equations, more difficult for solution. Expected in this case improvement of fit is rather not large. Presented result show that further investigations should be conducted for larger values of gas flow, probably larger than used here.

5 Conclusions

Following conclusion can be drawn on the basis presented investigations:

1. The theoretical model fits experimental results very well. It shows that the presented model of the process is correct. The fitting is better for larger gas velocity.
2. Gas in the pipes is perfectly mixed; in the column the flow approaches the plug, especially for larger gas velocity.
3. The main advancement of Laplace transform method application for solving such class of problems is its higher efficiency and convenience of calculations comparing to classical methods.
4. Using of CAS-type program (Maple®) significantly makes calculations simpler and faster.

Appendices

This work was supported by The National Centre for Research and Development (Poland) under a grant PBS1/A1/6/2012.

References

1. Widder D. V., 1946, *The Laplace transform,* Princeton University Press, USA.
2. Jaeger J. C., 1961, *An introduction to the Laplace transformation with engineering applications,* Methuen, London.
3. Poularikas A. D., 1996, *The transforms and applications handbook,* CRC Press, USA.
4. Kolev. S. D., Linden W. E., 1993, *Application of Laplace transforms for the solution of transient mass- and heat- transfer problems in flow systems,* International Journal of Heat and Mass Transfer, vol. 36, pp.135-139
5. Ahmed S., Batin A., 2013, *Convective laminar radiating flow over an accelerated vertical plate embedded in a porous medium with an external magnetic field,* International Journal of Engineering and Technology, vol. 3, pp. 66-72.
6. Ahmed S., Kalita D., 2012, *Laplace technique on magnetohydrodynamic radiating and chemically reacting fluid over an infinite vertical surface,* International Journal of Engineering and Technology, vol. 2, pp. 684-693.
7. Memberez J., Infelta P. P., Renken A., 1996, *Use of the Laplace transform technique for simple kinetic parameters evaluation. Application to the adsorption of a protein on porous beads,* Chemical Engineering Science, vol. 51, pp. 4489-4498.

DE GRUYTER
OPEN

JACSM 2014, Vol. 6, No. 1, pp. 55 - 65
DOI 10.2478/jacsm-2014-0004

EFFECTIVE APPROACH FOR DATA AGGREGATION IN WIRELESS SENSOR NETWORK: A STRUCTURE FREE APPROACH

Umashankar Prasad[1], Dr. D.S. Adane[2]

[1] Department of Computer Science and Engineering
Ramdeobaba College of Engineering and Management
Nagpur (M.S.) India
prasadud@rknec.edu

[2] Department of Information Technology
Ramdeobaba College of Engineering and Management
Nagpur (M.S.) India
hodit@rknec.edu

Abstract

Wireless Sensor Network consist of thousands sensor node which have limited Power, Computation, Sensing and Communication capabilities. Among all operation of Sensor Node, Wireless Communication consumes most of the energy. So it is necessary to decrease the number of packets transmitted through the network. Many Sensor Node could detect similar event, which increases the overall bandwidth utilization to transmit redundant data. Here Nodes computation is cheaper than communication in terms of energy. So the technique of Data Aggregation is applied to summarize data which decreases the amount of data transmitted in the network, which in turn increases the lifetime of the network. Many Data aggregation protocols are based on a structured approach which is suitable for data collection application. But maintenance of the structure is an extra overhead and this approach is not suitable for dynamic scenario. So we propose an ad-hoc data aggregation protocol for dynamic scenario mainly event based application.

Key words: Wireless Sensor Networks, throughput, delay, energy efficiency, protocols, event based, dynamic, Data Aggregation.

1 Introduction

Wireless Sensor Networks are composed of many inexpensive, low-powered sensing devices with limited computational, memory and communication resources [1], [2]. This networks offer various low-cost solution to

various problem including target tracking, environment and health monitoring, traffic regulation and wildfire detection. Since sensor node are low cost it had simple hardware with many resource constraint. Hence it is challenging to provide to gather data among these battery powered nodes, since node battery is limited. To reduce power consumption various methods like radio scheduling, topology control, control packet elimination and the most important data aggregation is used [2], [3]. Data Aggregation protocol aims at summarize data packets of different nodes so the total amount of data transmitted over the network could be minimized and bandwidth utilization also increases which in turn affects the through-put of the network.

Optimal Data Aggregation can be measured in terms of energy consumed by transferring the collected information to sink. But it heavily depends on the topology of network, placement of data aggregator nodes, event source, etc. Many centralized structured approach [4], [5], [6], [7], [8], [9] are mainly focused on data gathering application where all nodes report their sensor value periodical to Sink. Many distributed structured approach [10], [11], [12] are proposed for event based application. There are various limitation for using structured approach in event based application where event region change is frequent, First overhead of construction and maintenance of structure is not practical is dynamic scenario. Second absents of center of event source leads to in-optimal structure formation for aggregation. Third centralized computation of structure is not possible in dynamic scenario.

In this paper we focus on structure free data-aggregation mainly used for event based application. The purpose is to redesign DAA [13] (i.e. Data Aware Anycast) structure free data aggregation protocol to improve the lifetime of the network. There are two major challenges in structure free data aggregation. First, who should send data to whom for aggregation? Second, who should wait for whom before sending so optimal aggregation could be achieved? Fan [13] has divided the problem in two part spatial convergence and temporal convergence and had developed two approach to tackle it. DAA protocol tackles spatial convergence problem thereby solving the first routing challenge. RW (i.e. Randomized Waiting) protocol at application layer tackles temporal convergence solving second who should for whom challenge.

We propose a new model for Structure free data aggregation which combines the Mac-Layer protocol Data Aware Anycast (DAA) and Sensor Mac (S-MAC) and Application Layer protocol Randomized Waiting (RW). Based on observation combining the basic S-MAC protocol with DAA protocol improves the lifetime as compared with DAA [13] original protocol.

2 Related Work

In wireless sensor network, Nodes communication cost is far more expensive then the computation cost. Pottie and Kaiser [14] research said that energy involved in computing lots of instruction is equivalent to transmitting one packet over wireless channel, therefore data aggregation while routing the packets is very important to extend the lifetime of the network. Various protocol are proposed for in-network data aggregation purpose which are divided in broad category as Structure-Based and Structure Free.

2.1 Structured-Based Approach

In structure based approach there is construction of a network structure which enable the nodes to know which node should wait for whom before sending its own data so that node can aggregate data received from others. So there is an overhead of construction and maintenance of a global network structure which could be done centralized or distributed manner.

In cluster based network all sensor nodes are not homogeneous in terms of responsibility. Few nodes becomes cluster head and rest all remain sensor node. Every node belong to particular cluster and Each nodes send data to their sensor data to their respective cluster head who aggregates all the received data and send it to base station. Heinzelman introduced LEACH [15] protocol which has two stages of operation. First stage is setup phase in which whole network is organized is cluster and cluster head is selected and second stage is steady phase in which all nodes send data to cluster heads then cluster head aggregates data from all nodes and send aggregated packet to sink directly by high power transmitter. Cluster head is re-elected after some time interval since the high power transmission to sink consumes more. One may say that it should save more energy by aggregating then transmission to sink.

Lindsey introduced protocol PEGASIS [16] where the nodes organized themselves in chain pattern. The farthest node initiate the chain formation and then rest of neighboring nodes form a linear structure. While data gathering each node send data to neighbor node which aggregates it and send to next in linear fashion. Both LEACH and PEGASIS assumes that the sink is could be reached in one hop which poses some physical limitation on the network.

Tree-Based data routing protocol [17], [18] sensing information entropy for data compression (aggregation). They assume global knowledge of each nodes entropy information.

2.2 Structure-Free Approach

In this approach there is no need to maintain a global structure for data aggregation. Data aggregation done is such network may not be optimal when compared to structure approaches but the overhead of construction and maintenance is absent which is good for event based application where the event region changes frequently.

Major work done in this category is by Kai-Wei Fan. In this paper [13] author had observed the opportunity for data aggregation and had proposed DAA mac layer protocol and RW application layer protocol is do so. Which makes routing and aggregation decision dynamically.

Chao [19] have proposed SFEB protocol which extends the DAA approach and is suitable for specific network scenarios made improvements to it.

3 Spatial and Temporal Convergences

For optimal aggregation, all nodes should transmit in certain order. All structured data aggregation approach are designed to follow ordering of packets while forward it. Structured approach are well suited for data gathering application where every node has data to report and they form an hierarchy where leaves node send their data intermediate nodes wait and receives from all child packet then aggregates then forward it. These approaches have high overhead of maintenance of structure and are not suitable for event based application. For data aggregation in event based application is done only when the packets happen to meet at same node at same time with potential of aggregation. We study and design structure free approach for data aggregation well suited for event based application to avoid the overhead of maintenance of structure.

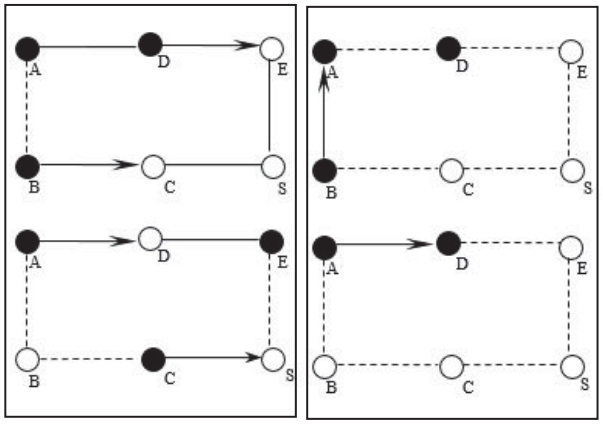
Spatial and temporal convergence are necessary condition for data aggregation. For structure free approach we would like to extend the work done by Kai-Wei Fan [13]. We would like to combine benefit S-MAC and DAA for greater efficiency and improved lifetime.

Main advantage of structure free approach are, First it is tolerance to dynamic events where the event region changes frequently. Second it is fault tolerance, nodes will aggregate and forward data till there is at least one node in its range. Third it is robust to mobility of nodes till it's in range of other nodes. Forth it is robust to interference, if there is failure while sending data it can retry in next cycle as long as node has power.

3.1 Spatial Convergence

In this section we are presenting the new scheme which combine S-MAC and DAA. For proper data aggregation global knowledge of few things are needed but that should be network topology. The protocol defined here need to have following assumption:

- Nodes need to know their own location and sink location. This could be done by localization protocol [20], [21] or GPS devices.
- Nodes are need to be time synchronized for coordinative sleeping required by S-MAC and for aggregating packets generated at same time or different parameter could be considered for aggregation of data.



(a) (b)

Figure 1. Showing an opportunity for aggregation. (a) Nodes without data aggregation send data along the route without checking the opportunity for aggregation. (b) Node got to know that one hop neighbor has opportunity for aggregation so send data along the path where aggregation is possible.

Figure 1a shows the scenario shows how data will be forward if the opportunity in aggregation is ignore and data is transmitted normally. Figure 1b show if node B knows that its hop neighbor node A has data that could be aggregated in future so node B forward data to node A instead of node C which happened in general condition.

Figure 2. Each duty cycle is divide in listening period and sleep period and listening period is synchronized with all node taking part in network.

First we are using S-MAC as lower mac layer in our protocol. S-MAC has an adaptive sleeping schedule which need synchronization which is one of our assumption. S-MAC has a duty cycle (DC) which is sub-divided into two parts listening period and sleeping period as show in figure 2. Duty cycle of all nodes are synchronized such that one can send data to its neighboring on its listening period only.

Data exchange takes place in a manner as RTS/CTS/DATA/ACK. For sending data across nodes RTS and CTS exchange must be happen in listening period so both sender and receiver know the data is coming then DATA/ACK could be exchanged over the sleep period and all the rest nodes not taking part in transmission can go to sleep as the listening period end. Here Duty cycle must be selected such that data transmission should be completed before the next listening slot begins. Carrier sense before sending any packet and the handshake model avoid the hidden and exposed terminal problem of wireless channel. Network Allocation Vector (NAV) could be send along the pack to alert other sensing nodes to back off certain amount of time.

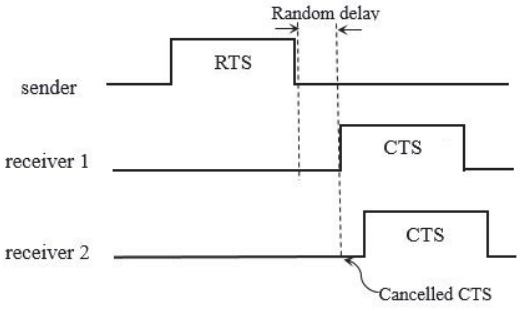


Figure 3. Showing how multiple CTS is cancelled

Second section of protocol is DAA based at MAC layer which decide the next hop for each transmission. It was RTS packet is decide which node is ready to receive transmission. RTS is send during the listening phase of S-MAC. The opportunity for aggregation in the next hop could be decided by Aggregation ID (AID). AID could be the timestamp of event so same event detected by multiple node could be aggregated or could be anything depending upon the applications. RTS packet contains AID within it and nodes re-

ceiving such RTS packet also has data with same AID so there is opportunity for aggregation. This way only those nodes with got AID match will reply with CTS. But the same RTS packet is received by more than one node and both the node reply to one RTS then there will be CTS collision to solve that. We delay the CTS with some minor random time to avoid that and channel is sensed first before sending CTS. So in this way subsequent is cancelled as shown in Figure 3. To increase aggregation. We will reply to RTS with CTS the receiving node is closer to sink then sender node.

Nodes are categorized into three classes and given priority. So higher priority nodes reply with CTS early then lower priority. These three classes of priority are as follows:

- Class A: Receiving node has same AID as in RTS and closer to sink then sender.
- Class B: Receiving node has same AID as in RTS and farther to sink then sender.
- Class C: Receiving node does not have has same AID as in RTS but it's closer to sink then sender.

Class A has more priority then Class B. Class B has more priority then Class C. So if a node receives a RTS it checks that to which class it belong and then reply CTS is delayed accordingly. So the class A send CTS first then other two class.

The information that each node have is the distance to sink which could be used as the routing parameter. This parameter in incorporated least priority in reply CTS for proper routing to sink.

3.2 Temporal Convergence

Aggregation only happens when two packet with same AID meet at same node and at same time. So if we transmit the data as soon as it is captured by sensor then some packets will be forward immediately and the chances of aggregation will be lost. Same AID of data generally exist among the data captured for same physical event by many nodes. If all nodes send data immediately by the above method there will be only few aggregation. So nodes need to wait for time before sending their own data so chances of aggregation is maximized. Since there is no structure this could be done in random manner. So random waiting (RW) at application layer is solution to it. Here nodes has a random wait period at the start of listening period before sending RTS to avoid collision as well as to solve temporal Convergence issue. However this method does not guarantee optimal aggregation due to its random nature but increases aggregation chances. This enables few nodes to wait few duty cycle before sending when they receive RTS from neighbor node before their own RTS gets ready.

The random value range selection plays crucial role in optimal aggregation that depends on the density of network, event size, time to transmit, listening and sleeping period chosen in S-MAC.

4 Simulation Scenario

To justify my design of the above protocol we need to compare it with non-aggregated and structured based approaches of data aggregation at least in simulation environment. We have chosen Ptolemy II simulator for justifying our design. Ptolemy II has all basic component for wireless network simulation and each and every component is customized from its wide actor set. It is an open-source software framework supporting experimentation with actor-oriented design. The model semantics in Ptolemy II is not dependent on framework but by a software component called director. It's developed and supported by UC Berkeley EECS dept. [22].

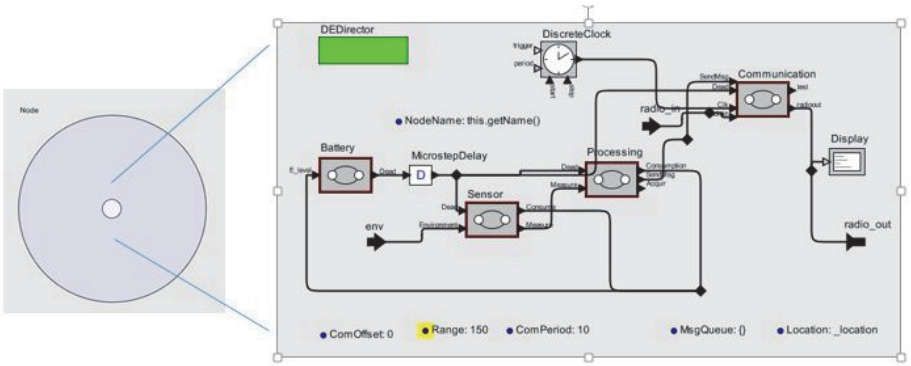


Figure 4. Sensor Node design in Ptolemy II.

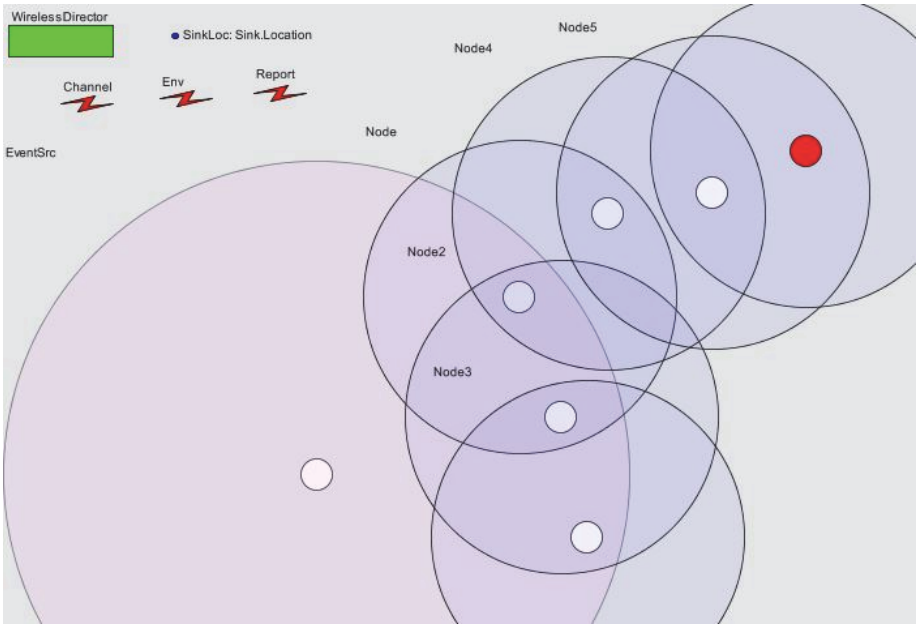


Figure 5. Simulation scenario for one event.

Each sensor node in our simulation is actor in wireless director environment. Sensor node is broken in four component as shown in figure 4. All four component design is based on finite state machine (FSM) where each state has refinements wherever needed.

A typical scenario is shown in figure 5 in which biggest circle on left is event source with its range. A sensor node is show with white circle with larger circle around it to represent the range that node covers. The node with red Center is sink node where the information need to be sent. Three nodes that fall in event region senses that event and try to report and broadcast RTS and the neighbor nodes in event region have higher priority as the AID could match reply with CTS early then not matching AID nodes outside the event region. So these three node aggregate the data among each other in three cycle and then multi-hop that one packet to red sink.

5 Future Work

The performance analysis of the above model need to done against the non-aggregated and structured based data aggregation approach. The report generation and analysis method are need to be incorporated in the proposed approach.

6 Conclusion

The lifetime of the network is maximized by introducing the S-MAC layer functionality over the DAA protocol. S-MAC basically solves the issue of idle listening of the wireless channel. So nodes has only to listen to channel in the listening cycle. This improvement over tradition DAA approach saves a lot more energy thereby increasing the lifetime. But it also increase per-hop delay, so there should be tradeoff between the delay and energy saved in this structure free aggregation method.

Acknowledgments

Editors of the JACSM would like to thank authors and reviewers for their interest in the subject of JACSM, contributions to the journal and their great job which enables to create contents of the journal's volumes.

References

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. Computer networks, 38(4), 393-422.
2. Yick, Jennifer, Biswanath Mukherjee, and Dipak Ghosal. "Wireless sensor network survey." Computer networks 52.12 (2008): 2292-2330.
3. Akkaya, Kemal, Murat Demirbas, and R. Savas Aygun. "The impact of data aggregation on the performance of wireless sensor networks." Wireless Communications and Mobile Computing 8.2 (2008): 171-193.
4. Kulik, Joanna, Wendi Heinzelman, and Hari Balakrishnan. "Negotiation-based protocols for disseminating information in wireless sensor networks." Wireless networks 8.2/3 (2002): 169-185.
5. Intanagonwiwat, Chalermek, Ramesh Govindan, and Deborah Estrin. "Directed diffusion: a scalable and robust communication paradigm for sensor networks." Proceedings of the 6th annual international conference on Mobile computing and networking. ACM, 2000.
6. Krishnamachari, Bhaskar, and John Heidemann. "Application-specific modelling of information routing in wireless sensor networks." Performance, Computing, and Communications, 2004 IEEE International Conference on. IEEE, 2004.
7. Heinzelman, Wendi Beth. Application-specific protocol architectures for wireless networks. Diss. Massachusetts Institute of Technology, 2000.
8. Lindsey, Stephanie, Cauligi Raghavendra, and Krishna M. Sivalingam. "Data gathering algorithms in sensor networks using energy metrics." Parallel and Distributed Systems, IEEE Transactions on 13.9 (2002): 924-935.

9. Ding, Min, Xiuzhen Cheng, and Guoliang Xue. "Aggregation tree construction in sensor networks." Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th. Vol. 4. IEEE, 2003.
10. Zhang, Wensheng, and Guohong Cao. "DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks." Wireless Communications, IEEE Transactions on 3.5 (2004): 1689-1701.
11. Zhang, Wensheng, and Guohong Cao. "Optimizing tree reconfiguration for mobile target tracking in sensor networks." INFOCOM 2004. Twenty-third Annual-Joint Conference of the IEEE Computer and Communications Societies. Vol. 4. IEEE, 2004.
12. Intanagonwiwat, C., Estrin, D., Govindan, R., & Heidemann, J. (2002). Impact of network density on data aggregation in wireless sensor networks. In Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on (pp. 457-458). IEEE.
13. Fan, Kai-Wei, Sha Liu, and Prasun Sinha. "Structure-free data aggregation in sensor networks." Mobile Computing, IEEE Transactions on 6.8 (2007): 929-942.
14. Pottie, Gregory J., and William J. Kaiser. "Wireless integrated network sensors." Communications of the ACM 43.5 (2000): 51-58.
15. Heinzelman, Wendi Rabiner, Anantha Chandrakasan, and Hari Balakrishnan. "Energy-efficient communication protocol for wireless microsensor networks." System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on. IEEE, 2000.
16. Lindsey, Stephanie, and Cauligi S. Raghavendra. "PEGASIS: Power-efficient gathering in sensor information systems." Aerospace conference proceedings, 2002. IEEE. Vol. 3. IEEE, 2002.
17. Scaglione, Anna, and Sergio Servetto. "On the interdependence of routing and data compression in multi-hop sensor networks." Wireless Networks 11.1-2 (2005): 149-160.
18. Scaglione, Anna. "Routing and data compression in sensor networks: stochastic models for sensor data that guarantee scalability." Information Theory, 2003. Proceedings. IEEE International Symposium on. IEEE, 2003.
19. Chao, Chih-Min, and Tzu-Ying Hsiao. "Design of structure-free and energy-balanced data aggregation in wireless sensor networks." High Performance Computing and Communications, 2009. HPCC'09. 11th IEEE International Conference on. IEEE, 2009.
20. Rudafshani, Masoomeh, and Suprakash Datta. "Localization in wireless sensor networks." Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on. IEEE, 2007.
21. Chraibi, Youssef. "Localization in wireless sensor networks." (2005).
22. The Ptolemy Project. http://ptolemy.eecs.berkeley.edu/.

DE GRUYTER OPEN

JACSM 2014, Vol. 6, No. 1, pp. 67 - 80

DOI 10.2478/jacsm-2014-0005

PERSISTENT SEQUENCES WITH EFFECTIVE RANDOM ACCESS AND SUPPORT FOR INFINITY

Konrad Grzanek

IT Institute, University of Social Sciences, Łódź, Poland
kgrzanek@spoleczna.pl, kongra@gmail.com

Abstract

Persistent sequences are the core data structure in functional programming style. Their typical implementations usually allow creating infinite streams of objects. Unfortunately, asking for length of an infinite data structure never ends or ends with a run-time error. Similarly, there is no default way to make an effective, *O*[*1*] or logarithmic access to an arbitrarily chosen sequence element, even when the nature of the correlation between index value and the sequence element is known. This paper presents a Clojure library that meets these limitations and offers an enhanced version of sequences with a support for effective random access and the ability to ask for an infinite length.

Key words: functional programming, Clojure, persistent collections, infinity, random access

1 Prerequisites for Enhanced Sequences

Sequences are by far the most common data structures in functional programming languages [3]. The name *Lisp* (originally *LISP*) derives from *LISt Processing*. In *Haskell* [5], [6] the list structure is also a basic non-atomic data type. Sequences possess a set of useful algebraic properties; they are *functors* and *monadic type constructors* [8]. Modern programming languages like *Clojure* [1], [2] and *Haskell* support non-strict evaluation This property of these languages makes creating infinite sequences natural. For instance in Clojure the expression

(***iterate*** *inc 0*)

and in Haskell, an equivalent form

iterate (1+) 0

of type

Num a ⇒ [*a*]

both denote an infinite sequence of natural numbers *0, 1, 2, 3, ...* In Clojure the function *clojure.core/iterate* is defined like

```
(defn iterate
  [f x] (cons x (lazy-seq (iterate f (f x)))))
```

and in Haskell

```
iterate :: (a → a) → a → a
iterate f x = x : iterate f (f x)
```

Moreover using *co-recursion* (see e. g. [4])the definitions like the infinite stream of values (*1* in this case)

ones = 1 : ones

or odd numbers

odds = 1 : map (+2) odds

are perfectly possible. In Clojure the lazy evaluation is not a default behavior, but the two co-recursive Haskell streams may be expressed like

```
(def ones (cons 1 (lazy-seq ones)))
```

and

```
(def odds (cons 1 (lazy-seq (map #(+ 2 %) odds))))
```

There is even a possibility to build a generalization of the emerging pattern using Clojure *macro*:

```
(defmacro lazy-cons
  [x seq]
  `(cons ~x (lazy-seq ~seq)))
```

and then one can write the expressions as
(***def*** *ones* (***lazy-cons*** *1 ones*))

(***def*** *odds* (***lazy-cons*** *1* (***map*** #(+ *2 %*) *odds*)))

This mimics the Haskell expressions closely, the differences are more on the syntactic than the semantic level with one clear semantic mismatch: the (:) operator in Haskell is a function of type $a \rightarrow [a] \rightarrow [a]$ and the *lazy-cons* introduced by us in Clojure is a macro that – as such – can't be passed around as a *first-class* value. This is still not an issue when defining streams co-recursively.

Infinite sequences (streams) are first-class citizens in the functional programming style under an assumption of non-strict (presumably lazy) evaluation. Although the examples given above make it clear, some questions still arise about querying these sequences for their length or about accessing their arbitrarily chosen element.

Let's introduce the symbol $\perp$ to depict a polymorphic "result" of a never ending computation and let the symbol $\Rightarrow$ mean "evaluates to" or "reduces to". The following are true

length ones $\Rightarrow \perp$

and

(*count odds*) $\Rightarrow \perp$

This is almost never a desirable computer program behavior. When we assume the infinity of some streams, it is natural to expect

length ones $\Rightarrow \infty$

and

(*count odds*) $\Rightarrow \infty$

Moreover, even in the face of operating on finite sub-streams, it is impossible to write

(*count* (*take* (** *2 1000*) *ones*))

and get a correct result, because

```
(defn count
  [coll] (clojure.lang.RT/count coll))
```

and

```
public static int count(Object o){
      if(o instanceof Counted)
            return ((Counted) o).count();
      return countFrom(Util.ret1(o, o = null));
}
```

The *count* method returns *int* values and so it can't give a *long* or (in this case) a *java.math.BigInteger* answer. In Haskell the function *length* is of type $[a] \rightarrow Int$, so it suffers the same kind of problems.

Another variant of these stream library inconveniences occurs with respect to a random access. In Clojure the *clojue.core/nth* procedure requires an *int* and in Haskell the operator (!!) has type $[a] \rightarrow Int \rightarrow a$. Moreover, the realizations of these operators in both languages take the same approach, like in the Haskell code below:

```
xs !! n | n < 0 = error ...
[] !! _         = error …
(x : _) !! 0    = x
(_ : xs) !! n   = xs !! (n-1)
```

This is why it is impossible to introduce some more effective ways to access the nth element in the sequence.

We aim here to present a Clojure library enhancement that solves the problems specified above. A similar, but not identical solution may be proposed for Haskell. The differences are caused by Haskell's static type checking and lack of sub-typing[1] ([6]).

2 Enhanced Sequences Implementation and Usage

The solution consists of a new sequence type that implements the following interface:

1 The Haskell solution requires introducing a specialized algebraic data-type with all interesting list operators, including the effective ones proposed in this paper, defined for it. Although it may seem disturbing at first, it is rather natural regarding the nature of Haskell type system.

```
public interface IEnhancedSeq extends
      IPersistentCollection, Sequential,
      Indexed, IHashEq {

  Number len();

  Object enth(Number n);
}
```

An enhanced sequence's length is represented here by a *len* operator of type *IEnhancedSeq* → *java.lang.Number*. This opens a way to return length values greater than *Integer.MAX_VALUE*. Similarly, the *enth* operator of type *IEnhancedSeq* → *java.lang.Number* → *Object* allows using *Numbers* instead of *int* values as indexes when accessing arbitrary sequence elements.

The enhanced sequences type implements Clojure interfaces common for predefined kinds of sequential types in the language, as presented at the above *IEnhancedSeq* interface definition.

The interfaces are implemented by a single abstract class *ESeq*. This section gives a detailed description of the class. The class definition goes as follows:

```
public abstract class ESeq implements List, IEnhancedSeq
{
  private final IPersistentCollection origin;
}
```

There are two constructor methods in the class. The one called *withLen* allows to bind a length generating procedure to the resulting enhanced sequence:

```
  public static Object withLen(final IFn len,
                               Object coll) {
    return new ESeq(origin(coll)) {
      @Override
      public Number len() {
        return (Number) len.invoke();
      }
    };
  }
```

The other one – *withEnth* – binds a random accessor:

```
public static Object withEnth(final IFn enth,
                              Object coll) {
    return new ESeq(origin(coll)) {
      @Override
```

```
      public Object enth(Number n) {
        return enth.invoke(n);
      }
    };
  }
```

After the enhanced objects are created they may be asked either for their length:

```
  @Override
  public Number len() {
    if (origin instanceof IEnhancedSeq) {
      return ((IEnhancedSeq) origin).len();
    }
    return origin.count();
  }
```

or for their nth element:

```
  @Override
  public Object enth(Number n) {
    if (origin instanceof IEnhancedSeq) {
      return ((IEnhancedSeq) origin).enth(n);
    }
    return RT.nth(origin, RT.intCast(n));
  }
```

In the two operators above a wrapped origin collection is used. In fact an *ESeq* is just a wrapper around the origin, one may say – a decorator – that offers the additional useful bindings established on the compile time.

The implementation works smoothly with the standard library mechanisms, because the original operators simply use the newly introduced enhancements, like

```
  @Override
  public final int count() {
    return RT.intCast(len());
  }
```

or

```
  @Override
  public final Object nth(int i) {
    return enth(i);
  }
```

and it's variant:

```
@Override
public final Object nth(int i, Object notFound) {
  try {
    return enth(i);
  }
  catch (IndexOutOfBoundsException e) {
    return notFound;
  }
}
```

An interesting point to be made here is the implementation of the operator that checks for the enhanced sequence emptiness. It uses a special *Infinity* type[2]. As it can be seen at the following listing, the *Infinity.POSITIVE* is an instance of *java.lang.Number* and it may be returned by the *len* operator:

```
private static final Long ZERO = 0L;

@Override
public final boolean isEmpty() {
  Number n = len();
  if (Infinity.POSITIVE == n) {
    return false;
  }
  return Numbers.equiv(n, ZERO);
}
```

This closes up the implementation details of *ESeq* class. There are many more mechanisms that have their place in the whole, but because their role is limited to ensuring the conformance with the standard library and it's contracts, we decided not to present them here.

On Clojure side there is one basic operator which answers whether or not the passed object (presumably a collection) is an enhanced seq or not:

```
(defn eseq?
  [coll]
  (instance? kongra.core.eseq.IEnhancedSeq coll))
```

2 Presenting the details of the *Infinity* type including the basic infinity-aware arithmetic operators lay beyond the scope of this paper.

2.1 Length and Infinity

The mechanisms described in the previous section are used to build another layer of abstraction, the one containing enhanced length-related operators. The most basic one allows to bind length generating function to the returned sequence (as mentioned above).

```
(defn with-len
  [len coll]
  (kongra.core.eseq.ESeq/withLen len coll))
```

There is also a possibility to pass a length value that gets bound immediately. The expression (*return* <*value*>) generates a function that always returns <*value*> and can be used as an argument to *with-len*:

```
(defn with-len-value
  [value coll]
  (with-len (return value) coll))
```

A useful macro *with-delayed-len* allows one to bind a lazily-evaluated length value:

```
(defmacro with-delayed-len
  [len-expr coll]
  `(let [d# (delay ~len-expr)]
     (with-len (return @d#) ~coll)))
```

and the operator *with-len-like* copies the binding for length from the origin into the resulting collection:

```
(defn with-len-like
  [origin coll]
  (with-len-value (len origin) coll))
```

Finally one can ask the collection for it's enhanced length value using the following *len* operator. It is worth noting that for collections of types other than *ESeq* simply *clojure.core/count* is used to establish the value:

```
(defn len
  [coll]
  (if (eseq? coll)
    (.len ^kongra.core.eseq.IEnhancedSeq coll)
    (count coll)))
```

As we stated in the previous section, the *len* operator may return the Infinity value. As a natural and desired consequence of this fact we may ask the collection if it is infinite or not. The following procedures simply compare the returned length value with Infinity constants:

```
(defn infinite?
  [coll]
  (+∞? (len coll)))


(defn finite?
  [coll]
  (not (infinite? coll)))
```

Two things should be underlined here:

1. If the collection asked for it is infinite length is not an enhanced one with len bound, then the count operator is used. This standard library function simply counts the sequence elements traversing it from the start. So the len operator is O[n] in the worst case.
2. No standard Clojure collection is an ESeq by default. This is why both hold: (infinite? (iterate inc 0)) ⇒ ⊥ and (infinite? odds) ⇒ ⊥

It is programmer's responsibility to mark a sequence as an infinite one. Fortunately, whe following simple procedure does the job.

```
(defn infinite
  [coll]
  (with-len-value +∞ coll))
```

Now with the following definitions

(***def*** *ones* (***infinite*** (***lazy-cons*** *1 ones*)))

(***def*** *odds* (***infinite*** (***lazy-cons*** *1* (***map*** #(+ *2 %*) *odds*))))

we have (*infinite? ones*) ⇒ *true*, (*infinite? odds*) ⇒ *true* and also (*len ones*) ⇒ ∞, (*len odds*) ⇒ ∞.

2.2 Random Access with Optimistic Performance Profile

Similarly, there are two operators, one that allows binding the effective random accessor procedure to the returned collection:

```
(defn with-enth
```

```
  [nth coll]
  (kongra.core.eseq.ESeq/withEnth nth coll))
```

and another that actually makes the access using the bound accessor, if present. In the face of absence of such an accessor binding, the standard *clojure.core/nth* is used:

```
(defn enth
  [coll n]
  (if (eseq? coll)
    (.enth ^kongra.core.eseq.IEnhancedSeq coll n)

    (nth coll n)))
```

3 Examples

To get a stronger grasp on what these enhancements may be useful for, please, take a look at the following procedure that concatenates collections and is aware of the possible infinity of some of the arguments:

```
(defn cat-eseq
  "Returns an enhaced collection being the result of concatenating the
  passed colls. It is assumed the number of colls is finite."
  [& colls]
  (let [;; prepare colls and lens (delayed)
        len-colls-bundle
        (delay
          (let [lens (map len colls)
                ;; take only the colls up to the first with
                ;; len=+∞ (inclusively)
                ∞-idx (find-idx' +∞? lens)
                colls (if ∞-idx (take (inc' ∞-idx) colls) colls)
                lens  (if ∞-idx (take (inc' ∞-idx) lens)  lens)]
            (pair lens colls)))

        lens  #(pair-first  @len-colls-bundle)
        colls #(pair-second @len-colls-bundle)

        ;; prepare enth (with delayed realization)
        intervs-intermap-bundle
        (delay
          (let [intervs (->> (lens)
                             (cons 0)
                             (apply cummulative-intervs')
                             vec) ;; essential wrt performance
                intermap (zipmap intervs (colls))]
            (pair intervs intermap)))

        intervs  #(pair-first  @intervs-intermap-bundle)
        intermap #(pair-second @intervs-intermap-bundle)

        enth-impl
```

```
        (fn [n]
          ;; 1. select proper interval
          (let [intv (binary-search (intervs) n
                                    #(interv-compare (lv "[, )") %2 %1))]
            (when-not intv (terror IndexOutOfBoundsException n))

            (let [ ;; 2. select collection
                  coll ((intermap) intv)
                  ;; 3. calculate the index in the collection
                  i (- n (:start intv))]
              ;; 4. get the result wrapped with the transformation
              (enth coll i)))))]

    (->> (apply concat (colls))
         (with-enth enth-impl)
         (with-delayed-len (reduce ∞+' (lens))))))
```

A slightly less complicated is the sequence of natural numbers. The first variant is not an enhanced one:

```
(defn N'
  "Returns an infinite N (natural numbers) set := 0, 1, 2, ...
  Optionally allows to specify the start (first) number. Unlimited
  integral range."
  ([]      (N' 0))
  ([start] (iterate inc' start)))
```

and now the enhanced version:

```
(defn N'-eseq
  ([] (N'-eseq 0))

  ([start]
     (->> (N' start)
          infinite
          (with-enth #(do
                        (when (< % 0) (terror
                           IndexOutOfBoundsException %))
                        (+' start %))))))
```

Similarly, the infinite sequence of factorial numbers, non-enhanced in the first place:

```
(defn factorials'
  "Returns an infinite stream 0!, 1!, 2!, 3!, ..."
  []
  (iterate-with *' 1 (N' 1)))
```

and it's enhanced version:

```
(defn factorials'-eseq
  "An eseq version of factorials'"
  []
  (->> (factorials')
       (with-enth #(factorial' %))
       infinite))
```

The *power-set* (set of all subsets) implementation takes a slightly more composite approach. First, the generator:

```
(defn- powerset-generator
  [indexed-coll n]
  (->> indexed-coll
       (take (ebit-length n))
       (filter (fn [p] (ebit-test n (pair-first p))))
       (map pair-second)))
```

then, the accessor

```
(defn nth-in-powerset
  "Returns an n-th element of a powerset of a collection. Works for
  n : T <: Long and for (possibly) infinite collections. That's why
  for the finite colls there are no range checks for n."
  [coll n]
  (powerset-generator (indexed' coll) n))
```

And the actual power-set sequence:

```
(defn powerset
  "Returns a powerset for a possibly infinite coll."
  [coll]
  (let [indexed-coll (indexed' coll)]
    (->> (N' 1)
         (map #(powerset-generator indexed-coll %))
         (take-while seq)
         (cons '()))))
```

The enhanced power-set sequence takes an additional element transformation function, called *enthtrans*, to (optionally) modify any value either when accessing it during a standard iteration (e.g. left or right *catamorphism* – see [7]) or by an accessor:

```
(defn powerset-eseq
  "Returns an enhanced version of the powerset. Every element of the
  returned collection is subjected to a transformation using enthtrans
  (identity by default)."
  ([enthtrans coll]
     (let [n (len coll)
           result (->> coll
                       powerset
                       (map enthtrans)
                       (with-enth #(enthtrans (
                                    nth-in-powerset coll %))))]
       (if (+∞? n)
         (infinite result)
         (with-delayed-len (** 2 n) result))))
  ([coll]
     (powerset-eseq clojure.core/identity ;; enthtrans
                    coll)))
```

The final case-study is an enhanced permutations generation routine. The accessor returns *n*-th permutation of a collection of elements using *Lehmer code* for *n* [9]:

```
(defn nth-permutation
  "Returns n-th permutation (lexicographical) of the given coll."
  [coll n]
  (let [v (if (vector? coll) coll (vec coll))
        lehmer-code (factoradic n)

        ;; lehmer-code must be supplemented with 0-s to match the
        ;; length of v
        zeros-count (- (count v) (count lehmer-code))
        lehmer-code (concat (repeat zeros-count 0) lehmer-code)

        gen (fn [[_ v] i] (pair (nth v i) (vec-remove i v)))]

    (->> (iterate-with gen (pair nil v) lehmer-code)
         next
         (map pair-first))))
```

The original lexicographical permutations are returned by *clojure.math.combinatorics/permutations* routine. This is wrapped within the following enhanced form:

```
(defn permutations-eseq
  "Returns an enhanced version of permutations. Every element of the
  returned collection is subjected to a transformation using enthtrans
  (identity by default)."
  ([enthtrans coll]
     (->> coll
          permutations
          (map enthtrans)
          (with-delayed-len (factorial' (len coll)))
          (with-enth #(enthtrans (nth-permutation coll %)))))

  ([coll] (permutations-eseq clojure.core/identity ;; enthtrans
                             coll)))
```

Here we also have an optional element transformation (*enthtrans*), like in the case of the power-set.

4 Conclusions

We presented a set of convenient extensions for the sequence abstraction in Clojure. The attached case studies show the relative ease of using these mechanisms. In general, the enhanced sequences fit pretty well in the ecosystem Clojure standard library. However, it would be an instructive experience to implement them in statically typed languages like Haskell. The Clojure solution is based on sub-typing, which is typical for a language that compiles down to Java byte-code and runs on top of the JVM. In Haskell there is no sub-typing, so the expected implementation technique presumably should consist of:

– using the algebraic data types
– using type-classes
– and finally – the resulting enhancement abstraction should make it's usage explicit rather than implicit as in the case of Clojure.

As a reward, one would get a statically typed, provably correct solution. It is now a question of undertaking future efforts to make this happen.

References

1. Halloway S., 2009, *Programming Clojure*, ISBN: 978-1-93435-633-3, The Pragmatic Bookshelf
2. Emerick Ch., Carper B., Grand Ch., 2012, *Clojure Programming*, O'Reilly Media Inc., ISBN: 978-1-449-39470-7
3. Bird R., Wadler P., *Introduction to Functional Programming*, 1988, Prentice Hall International (UK) Ltd
4. Doets K., van Eijck J., *The Haskell Road to Logic, Math and Programming*, 2004, College Publications, ISBN-10: 0954300696, ISBN-13: 978-0954300692
5. Lipovaca M., *Learn You a Haskell for Great Good*, 2011, ISBN: 978-1-59327-283-8
6. *Haskell Wikibook*, 2014, http://en.wikibooks.org/wiki/Haskell
7. Meijer E., Fokkinga M.M., *Functional Programming with Bananas, Lenses, Envelopes and Barbed Wire*, 1991, Springer Verlag
8. Awodey S., *Category Theory, Second Edition*, 2010, Oxford University Press
9. Lehmer D.H., *Teaching combinatorial tricks to a computer*, 1960, Proc. Sympos. Appl. Math. Combinatorial Analysis, Amer. Math. Soc. 10: pp. 179–193

DE GRUYTER OPEN

JACSM 2014, Vol. 6, No. 1, pp. 81 - 87

DOI 10.2478/jacsm-2014-0006

THE ROLE OF INFORMATION TECHNOLOGY FOR PEOPLE WITH INTELLECTUAL DISABILITIES

Alina Marchlewska[1], Marek Gębarowski [2], Piotr Goetzen[3]

[1]IT Institute, University of Social Sciences, Łódź, Poland
amarchlewska@spoleczna.pl

[2]IT Institute, University of Social Sciences, Łódź, Poland
mwgebarowski@gmail.com

[3]IT Institute, University of Social Sciences, Łódź, Poland
goetzen@spoleczna.pl

Abstract

A mentally disabled person has limited access to information and communication. Making a computer available to such a person we give them an opportunity to be independent in life and develop their self-reliability. They can take part in various forms of social life, develop artistically, enrich their personality and widen the scope of interests. The possibility to use a computer can also be a way of spending free time.

Key words: IT qualifications, life-long learning, digital and social divide, intellectual disabled person.

1 Introduction

A computer with access to the Internet is now an integral part of each day for Polish people. It is a tool of work, study and entertainment. One can find a job, learn, make and keep contacts, get information about interesting social and cultural events, develop hobbies and interests, arrange administration matters on the Internet. Disability can be inborn, genetic, or a result of an illness or injury. Everybody can become disabled regardless of their gender, age, place of living, education or lifestyle.

Nevertheless, it is still one of the most discriminated social groups. The disabled struggle with many problems in their everyday life. These are physical problems (architectonic barriers, difficult transportation, communication), mental problems (low self-esteem, bad mood, difficulty in making contacts) or economic problems (unemployment, low salary,

treatment costs). In the larger socio-cultural context the disabled face problems connected with a lack of proper legal regulations, social isolation, limited or difficult access to various goods (information, education, specialistic medical help) [7].

One can often meet a stereotypical point of view that the disabled do not need to use modern technologies as they are not able to operate them and because this equipment is expensive as it must be adjusted to the needs of a disabled person [6]. A computer and the Internet have been used as tools enhancing education and rehabilitation of disabled youth, e.g. to stimulate cognitive functions and to eliminate developmental disorders.

A lot of research has shown that mentally disabled children can quite quickly master the basics of the computer and simple programs and they do it willingly. If a computer is used for rehabilitation purposes with disabled children, it is highly probable that they will be able to operate it better as adult people [8].

Mental disability is a big difficulty for an ill person. The kind of the illness can make the educational process difficult or even impossible. The symptoms of the illness (e.g. poor concentration, psychotic symptoms, cognitive process malfunction) can be the reason of terminating education due to difficulties in comprehending information. Another reason for not continuing education can be the fact that the disabled person is embarrassed with their illness or is discriminated by their peers.

2 Using a computer while working with children disabled mentally

Traditional prophylactic and re-educational methods with children disabled mentally are at times ineffective and poorly adjusted to new and changing life conditions. Modern therapeutic treatment should be a holistic approach to a child. It should stimulate their cognitive interests, develop creativity and teach making contacts with people.

Thanks to their many advantages as for animation and stimulation of processes and physical, chemical, biological phenomena invisible to eyes or very difficult to see in real conditions, computers are an attractive didactic means. They stimulate the following functions: cognitive, emotionally-motivational, practical, consolidating and controlling.

The computer influences a lot of senses allowing a teacher not only to cross the border of verbal communication but also to engage students emotionally by combining cognitive values with the aesthetic ones – text in connection with music, graphics or film. The computer combines features of many traditional devices used for recording, presenting, processing and sending information. Computer software allows one to consider individual

differences of students. Well thought-over computer use in educational processes gives the teacher a real chance of introducing invaluable changes in achieving educational goals.

To improve all, or almost all, distorted areas one can use computer techniques which can function in three ways:

1. cognitive and educational – computer programs allow people of poor intellectual ability to see and learn about the world, which in turn improves distorted functions.

2. emotional and motivational – the attractiveness of computer programmes impacts emotions making the effects of learning continuous. Computer techniques that are chosen and realised accordingly influence the emotional side of the student and enhance their motivation. Computer techniques activate the student's cognitive motivation and this in turn improves the educational process. Computer education keeps the educational tension providing new and various stimuli.

3. inter-communication – creates a possibility to communicate in the case of children with difficulties in making contacts with people. It allows them to increase the number of social contacts through interests common for healthy children, e.g. games, software. It refers to both children disabled mentally and children with another disability, e.g. in motion or hearing.

A student starting to work with a computer learns its secrets and computer programs directly and indirectly improving the abilities of reading and writing. They learn new terms connected with the way the computer functions in and the whole process is natural by using a mouse and a keyboard.

Working with the computer requires visual-auditory coordination, which also guarantees manual skills. Graphic and drawing programs are extremely invaluable as they improve visual-motor function, or a child's psychomotorics. However, they make the child concentrate very hard. Drawing with a computer requires mental involvement which results from the fact that computer graphics stimulates creative thinking. Computer painting improves exploration processes, requires perception, reception and information and processing it. The computer, thanks to graphics, animation and the possibility to present a still and a moving picture, stimulates imagination. Working with the computer, especially with a mouse, requires logical thinking. Painting or drawing is a visible effect of its work. The main advantage of using a computer is much longer time of concentration. Hyperactive children who cannot concentrate for a long time while working with a computer are able to focus much longer. It results from the fact that the computer is a very attractive work tool. It is not associated with any stressful situation in children's minds and it does not pose any danger. The patience of the device

encourages repetitions and new attempts to solve a task and an incorrect solution does not result in punishment [9].

3 IT qualifications vs employment possibilities for people disabled mentally.

Work is very important in the life of an adult man and its lack has negative effects to self-esteem and mental state, which in the case of the disabled, who find it hard to get or keep a job, is even more destructive to their functioning. Mental disability often makes a person withdraw from socializing. The illness limits autonomy and independence. In the case of mental disability the help of third parties or institutions is necessary therefore it is vital to minimise the negative effect of disability and use all available forms of therapy and rehabilitation [5].

Using information and communication technologies has an invaluable meaning while looking for a job. Knowledge is currently one of the biggest assets, especially in the professional area. A lot of information in its basic form can be obtained thanks to the Internet access.

Due to the fact that the Internet and related technologies are widely used, they are becoming part of almost all areas of socio-economic life. Corporations are trying to use available technologies in the most effective way, which means widening knowledge and skills in this area by employees and those who are looking for jobs. Public administration is also trying to adjust to the changing requirements of the citizens through the so called e-offices. Unlimited possibilities to purchase goods and services also influence the change of the consumer behaviour patterns. Due to the speed of change in the information society a man has to continuously develop professionally and in their personal qualifications.

4 Upgrading professional qualifications

In our dynamically changing time, continuous education, getting new skills, and upgrading qualifications are necessary conditions of functioning both professionally and socio-culturally. One of the conditions to achieve this is access to knowledge, information and various forms of education.

E-learning, i.e. distant learning with the Internet, has many advantages and can be a good solution for disabled people. People with motoric disability, who face various architectonic barriers, wanting to take part in educational classes can benefit from this method of learning. E-education which in fact is life-long learning can offer a lot to the disabled provided that educational materials and websites are prepared especially for them [10].

It is worth mentioning that thanks to the European Union funds the number of free trainings on computers, software, e-marketing, etc. dedicated to the disabled has risen greatly. Many non-government organizations and institutions working for the disabled can see the need to develop skills connected with computer use to prevent social and professional exclusion of this group. Lack of according educational actions, access to equipment and software, is one of the reasons for the growing digital divide of the disabled.

In the continuously changing reality the condition to function independently is widening your knowledge continuously, upgrading qualifications, and acquiring new skills. Using the Internet helps to find and keep a job, decreases the risk of unemployment, increases the chances to find a job when one do not have it. Nowadays, without knowing how to use a computer and the Internet it is very difficult to find a job as the majority of job offers are available only online. In the job offers requirements there is at least basic knowledge of how to use a computer and in the case of the disabled, who physical work is usually impossible for, computer skills can be an advantage or necessity while working from home [3].

5 IT tools preventing social divide

According to recent surveys people undergoing psychiatric or psychological treatment are still one of the social groups in Poland that feel discriminated [4]. It is particularly noticeable among people up to forty years old.

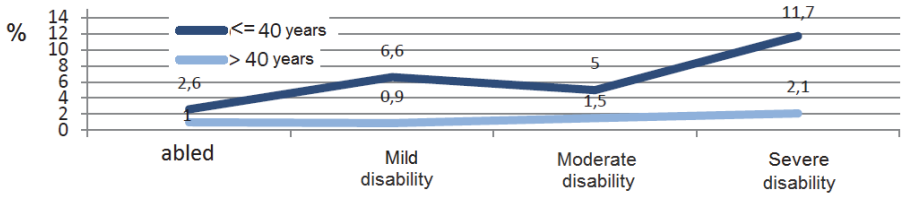


Figure 1. Percentage of people who feel discriminated according to their disability status and age [3].

People who suffer from mental disorders experience fear of being socially stigmatized. They often withdraw from socializing, have difficulties with functioning professionally due to the illness symptoms as well as being hospitalized repeatedly.

In Polish society there is still a negative stereotype of the disabled, which impacts the feeling of being stigmatized, rejected and lonely. In this case a computer can be used in many ways. Thanks to access to the Internet it is

possible to gain knowledge on the illness, read stories of other ill people, get support, find specialistic theurapeutic or medical help in dealing with the illness. A mentally ill person, or disabled in any other way, can make contacts without being labelled as ill by using the Internet . A computer can be also helpful in developing passions or hobbies as well as in education. Developing one's interestes and acquiring new skills is theurapeutic, increases self-esteem and makes the feeling about oneself better, which is immensely helpful in dealing with the illness.

There are many social portals for the disabled – these are niche portals dedicated to a given group of people with certain disability, e.g. http://www.bardziejkochani.pl/ - for families and caretakers of people with Down syndrome or http://www.ofon.net/ - Polish forum for the disabled.

Summary

Mental disorders, eyesight or hearing impairment require according conditions and specialistic software for training on how to use a computer. Therefore it is vital that a disabled person knows at least the basics. It may be a starting point for using various programmes for learning (reading, drawing, speaking a foreign language) by ill people. However, one should remember that as for using new IT technologies patients who have been mentally ill require special conditions and educational techniques concerning the nature of their disability [2].

While discussing the beneficial influence of communication and information technologies on the disabled one should mention the following aspects: information and cognitive, educational, cultural, rehabilitating, revalidating and integrational. With an according choice of tools (e.g. educational games) it is a good place of entertainment and relax.

One should not forget the factors which can be a greater danger to the disabled (especially intellectually) than to healthy people. One should remember the fact that the disabled are more prone to media manipulation (consumer media mainly), unable to analyze ambivalent, unclear or polyvalent content. Another danger can be the fact that people with mental disorders have difficulties in telling fiction from reality and recognizing situations, places and time.

Unequal access to IT technologies can result in far-reaching consequences. Nowadays, they are so common in many areas of life that people who do not use them are at risk of social divide. It is the so called digital divide.

One should stress the fact that in the case of the disabled the problem of digital divide – no possibility to use IT technologies - can be even bigger due to the kind of the disability. Many disabled people (e.g. blind, deaf, with

explorative and physical disorders) need additional equipment or special software, which is expensive and difficult to obtain, to use a computer.

In rehabilitation of the disabled the holistic approach appears to be necessary. Medical aspects will be equally important to improving the mental state of the disabled. They need to be self-reliant, adapt to the environment, get better education, be prevented from professional divide. The society should be educated on disability, which in return can help cross psychological, social, structural or legal barriers. One of the ways to achieve it is giving the disabled full access to IT technologies and showing them various possibilities to use them.

Nevertheless, one should remember that IT technologies are not a fantastic cure for all the problems of the disabled.

References

1. Batorski D., *Relacja wykluczenia społecznego z wykluczeniem informacyjnym,* Uniwersytet Warszawski, 2008.
2. Bronowski P., Sawicka M., Kluczyńska S. *Zastosowanie internetu w przełamywaniu niepełnosprawności psychicznej. Doświadczenia własne*, Psychoterapia 3(150), 2009.
3. Czapiński J., Panek T., *Social Diagnosis 2013, objective and subjective quality of life in Poland,* Rada Monitoringu Społecznego, Warszawa 2013.
4. Dykcik W. (red.), *Pedagogika specjalna,* WNU im. Adama Mickiewicza w Poznaniu, Poznań 2006.
5. Frąckiewicz L. (red.), *Przeciw wykluczeniu osób niepełnosprawnych*, wyd. IPiSS, Warszawa 2008.
6. Gorajewska D., *Fakty i mity o osobach z niepełnosprawnością,* Stowarzyszenie Przyjaciół Integracji, Warszawa 2009.
7. Karczmarek G., Karolińska B., Kruczek A., Płatek I., Polak M., Sobkowiak M. *Standard pracy socjalnej z osobą z niepełnosprawnością i jej rodziną z uwzględnieniem osób z zaburzeniami psychicznymi*, Standard w pomocy, 2009.
8. Siemieniecki B. *Rzeczywistość wirtualna w procesie wspomagania rozwoju i życia człowieka niepełnosprawnego*. [w:] *Człowiek niepełnosprawny - Rodzina i praca,* red. B. Aouil i M. Kościelska, WU AB, Bydgoszcz 2004.
9. Społeczeństwo informacyjne w Polsce. Wyniki badań statystycznych z lat 2007-2011.", GUS, US w Szczecinie, Informacje i Opracowania Statystyczne, Warszawa 2012.
10. Ślusarczyk Cz., *Rola Internetu w edukacji osób niepełnosprawnych*, Szkoła Główna Handlowa w Warszawie, 2006.