ATLANTIS PRESS
Advances in Intelligent Systems Research (AISR), volume 151
2018 International Conference on Computer Modeling, Simulation and Algorithm (CMSA 2018)

Using GMM-HMM Model and Parallel Computing for Health Estimation and Prognosis of Turbofan Engines

Zilu Wang
Aero Engine Academy of China, China

***Abstract*—Estimating the residual useful life (RUL) time of turbofan engines is a determinant factor for aviation safety. Different methods of machine learning are explored in the field of turbofan health estimation and prognosis, to save maintenance cost and enhance aviation safety. An algorithm used for turbofan engines residual user life estimation called gaussian mixture model and hidden Markov model is one of the most effective method many researchers used in complex system health estimation and prognosis. However, Gaussian Mixture Model and Hidden Markov Model (GMM-HMM) has N2 computational complexity, which is always time consuming under the condition of large dataset is available. During past two decades, along with more and more advanced sensors are used for turbofan engine health monitoring, since a large amount of data is available for observation and analyzation. Parallel computing library such as CUDA is extremely necessary. In this paper, CUDA library is adopted for turbofan engines RUL estimation. The performance of algorithms is demonstrated on a simulated aviation dataset generated by the Commercial Modular Aero-Propulsion System Simulation (CMAPSS).**

Keywords—GMM; HMM; Aero-enigine; RUL

I. INTRODUCTION

RUL prognostics is a fundamental factor to perform condition-based maintenance. Over the past decades, some studies have been conducted to algorithm for RUL prognostics of aero engine, to enhance aviation safety and decrease maintenance cost. During 70s and 80s of the last century, Britain's ministry of defense deployed turbo-fans life detector on AV8 fighters, to enhance logistic and supply chain performance. A common turbo fan engine has its core components, which include fan, compressor, combustor and turbine, also named gas path components. Due to gas path components are always working in high pressure and high temperature condition, gas path components fault is main source of turbofan engine failure [1]. At present, RUL estimation methods can be divided in two different types, physical model prognostics and data driven prognostics [2,3]. Physical model-based methods typically need expertise and building mathematical models to depict physical relations between subsystems. Although physical model-based methods have higher accuracy, however, owing to high complexity and strong interaction between subsystems of aero engine, they are always difficult to determine degradation propagation model. Data driven prognostics always utilize artificial intelligence and data mining methods to analysis and estimate aero engine performance degradation, finally to forecast RUL through building fault decision model. Artificial neural network is one of the most popular data driven technique in the prognostic algorithms. Chunyan Ma [4] combined gray model and neural network, trained by historical faulty data, to forecast engine failure. An adopted neural network method has been used in aero engine performance and wear trend estimation by Lvwei Yang [5]. In paper [6] a recurrent wavelet neural network is developed to predict rolling element bearing crack propagation. [6] proposed an algorithm using Bayesian model and Least Square SVM to forecast RUL based on the turbofan engine test data. The use of Hidden Markov Models in bearing fault prognostics in [7].

In present work, GMM-HMM is used to estimate the RUL of a turbofan engine, the HMM uses 5 output parameters from CMAPSS as observation inputs probably density functions. Dataset are generated by CMAPSS simulator, which include 28 parameters. This paper consists of five sections. Section two describes the CMAPSS dataset and data -preprocessing. Section three gives description about HMM and GMM. In section four the algorithms based on CUDA parallel computing is presented and in section five the simulation results and analyzation are shown. Finally, section five refers to comments and the future work.

II. PROBLEM DESCRIPTION AND DATA PRE-PROCESSING

To ensure flight safety, aircraft engine discs are usually retired at the time when one out of every thousand discs has a detectable damage. This means that over 99.9% expensive turbine discs are retired before their real life. Thus, research in a condition-based predictive maintenance is receiving more attentions with objects of maximizing aviation safety, minimizing operational risks, and reducing costs. Based on Commercial Aviation Propulsion Simulate Software (CMAPSS) developed by NASA Ames center, some researchers i.e. Saxena gives a series turbine fan engines fault simulation.

The data provided is from a high-fidelity system level engine simulation designed to simulate nominal and fault engine degradation over a series of flights. The flights are full flight recording sampled at 1 Hz and consists of 30 engine and flight condition parameters. Each flight contains 7 unique flight conditions for an approximately 90 min flight including ascent to cruise at 35K ft and descent back to sea level.

To eliminate impact caused by difference fan inlet condition, all data are transformed to fan inlet standard level, etc. fan inlet temperature is 255.15K, pressure is 101.325Kpa. The transformation formulas are listed below (Table 1):

TABLE I. FAN IMPACT ELIMINATION

Temperature transformation	$T_{cor} = T_m \frac{288.15}{T_2}$
Pressure transformation	$P_{cor} = P_m \frac{101.325}{P_2}$
Fan speed transformation	$n_{cor} = n_m \sqrt{\frac{288.15}{T_2}}$
Fuel flow transformation	$wf_{cor} = wf_m \frac{101.325}{P_2} \sqrt{\frac{288.15}{T_2}}$
Coolant bleed transformation	$W_{cor} = W_m \frac{101.325}{P_2} \sqrt{\frac{T_2}{288.15}}$

III. GAUSSIAN MIXTURE MODELS-HIDDEN MARKOV MODELS AND ESTIMATION PROCEDUR

A Hidden Markov Model is a statistical Markov model in which the system being modeled is assumed to N be a Markov process with unobserved states. Hidden Markov Models are especially known for their application in supervised learning and pattern recognition such as speech, handwriting, gesture recognition[8]. States space is defined as $\{S_1, S_2, S_3 \cdots, S_N\}$, with N denotes state number, and then define the HMM as $\lambda(A, B, \pi)$ where:

- $A = \{a_{i,j}\}$ is state transition matrix, $a_{i,j} = P\{q_{t+1} = S_j | q_t = S_i\}$.
- $\pi = \{\pi_i\}$ is initial state distribution vector, $\pi_i = P\{q_0 = S_i\}$.
- $B = \{b_i\}$ is the output probability density function, $b_i = P\{X|S_i\} = \sum_{m=1}^{M} c_{i,m} N(x; \mu^k, \Sigma^k), 1 \le k \le M$.

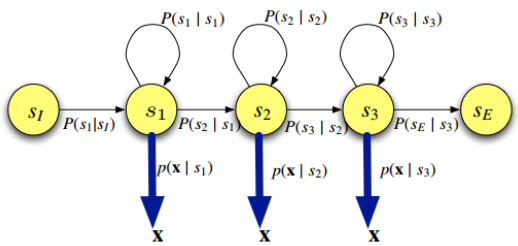


FIGURE I. FAULT PROGRESSION PROCESS DESCRIBED BY HMM

An observation X is conditionally independent of all other observations given the state that generate it. A state is independent of all other states given previous state. $1 < i, j < N$, $1 < t < T$, output datasets is clustered by m-components Gaussian mixture models and M is the number of mix factors, and it is satisfy $\sum_{m=1}^{M} c_{i,m} = 1$, and $0 \le c_{i,m} \le 1$, $c_{i,m}$ is the weight coefficient of the components. Expectation Maximization (EM) algorithm is selected for GMM parameters estimation. Back to HMM problems, the three problems of HMM are:

Likelihood Determined the overall likelihood of an observation sequence $\mathbf{X} - (x_1, \cdots x_n, \cdots, x_N)$ being generated by an HMM.

Decoding Given an observation sequence and an HMM, determine the most probable hidden state sequence.

Training Given an observation sequence and an HMM, learn the best parameters $\lambda(A, B, \pi)$.

GMM-HMM is trained by EM method and Baum-Welch algorithm. The URL estimation step is using Viterbi method to estimate current state by observation data, and then Monte Carlo algorithm is used for calculating how many flights left before changing to next state.

IV. COMPUTATION IN COMPUTE UNIFIED DEVICE ARCHITECTURE

Corresponding to the three problems of HMM model, three solutions are Forward algorithm, Viterbi algorithm and Baum-Welch algorithm respectively. The Viterbi and Baum-Welch Algorithm has a N^2 computational complexity, and simulator signals generates observative data sampled at 1 Hz for one flight. CUDA is a parallel computing platform and application programming interface (API) model created by Nvidia. It allows software developers and software engineers to use a CUDA-enabled graphics processing unit (GPU) for general purpose processing.

A. *Data Preprocessing*

Each thread is responsible for a batch of data samples transformation, which are allocated to GPU threads as in Table.2.

TABLE II. DATA PRE-PROCESSING PSEUDO-CODE

Data Pre-processing	
1.	Thread_id = threadIdx.x + blockIdx.x * blockDim.x
2.	WHILE Thread_id < N
3.	Data transformation calculation (in Table.1)
4.	Thread_id += blockDim.x * gridDim.x
5.	END

B. *Gaussian Mixture Model Cluster*

Initialize $c_{i,k}, u^k, \Sigma^k$ on global memory, in each iteration we calculate $\gamma(Z_k)$ on GPU, and $c_{i,k}, u^k, \Sigma^k$ are updated in parallel on GPU by using reduce sum and reduce mean operation.

TABLE III. GMM CLUSTER PSEUDO-CODE

GMM cluster
6. Set the parameter M, initialize $c_{i,k}, \mu^k, \Sigma^k$.
7. For each observation sample, calculate posteriori probability $\gamma(Z_k) = p(Z_k = 1\|\mathbf{x}) = \frac{c_k N(\mathbf{x}\|\mu_k, \Sigma_k)}{\sum_{m=1}^{M} c_m N(x;\mu^k, \Sigma^k)}$.
8. Update μ_k with the formula $\mu_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})x_n$. Where $N_k = \sum_{n=1}^{N}\gamma(z_{nk})$.
9. Update Σ_k using maximum likelihood function $\Sigma_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T$.
10. Solve maximum likelihood function of c_k, and update c_k by $c_k = \frac{N_k}{N}$, where $N_k = \sum_{n=1}^{N}\gamma(z_{nk})$.
11. Repeat procedure 2-5 until convergence.

C. *Forward Algorithm and Viterbi Algorithm*

For forward algorithm, forward probability $\alpha(t)$ are calculated in parallel. Each $\alpha(t,i)$'s calculation (Line 6 in Table.2) is allocated to GPU threads, and shared memory is used for sum reducing operation. Viterbi algorithm's parallel strategy is the same as forward algorithm, the only difference is change the sum to the max operation.

TABLE IV. FORWARD ALGORITHM PSEUDO-CODE

Forward algorithm and Viterbi algorithm
1. Initialize all $\alpha(t,i)$ to 0, $1 \le t \le T, 1 \le i \le N$
2. $\alpha(1,i) - \pi_i b_{i,o1}, 1 \le i \le N$
3. FOR $t = 2; t \le T; t++$ DO
4. FOR $i = 1; t \le N; i++$ DO
5. FOR $j = 1; j \le N; j++$ DO
6. Calculate $p = a_{ji} \times b_{i,ot}$ and $\alpha(t,i) = \alpha(t,i) + \alpha(t-1,j) \times p$
7. ENDFOR
8. ENDFOR
9. ENDFOR

D. *Baum-welch Algorithm*

Forward probability α and backward probability β are calculated in parallel in GPU threads, and using shared memory to calculate mean value for each state (Line 5) in parallel reducing.

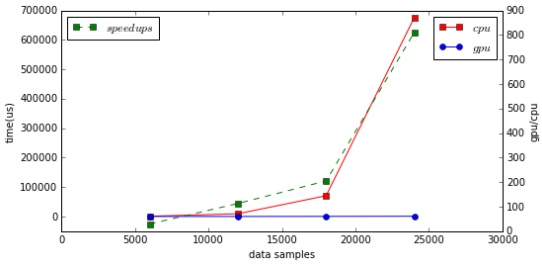
TABLE V. BAUM-WELCH PSEUDO-CODE

Baum-Welch algorithm
1. initialize α, β, γ and ε to 0
2. using forward algorithm to calculate $P(O\|\lambda)$
3. $\beta(T, i) = 1, \ 1 \le i \le N$
4. FOR $t = T - 1; t \ge 0; t--$ DO
5. FOR $i = 1; i \le N; i++$ DO
6. Calculate $\gamma(t,i) = \gamma(t,i) + \alpha(t,i) \times \beta(t,i)/P(O\|\lambda)$
7. Calculate $\varepsilon(i) = \varepsilon(i) + \alpha(t,i) \times \beta(t,i)/P(O\|\lambda)$
8. FOR $j = 1; j \le N; j++$ DO
9. Calculate $p = a_{ji} \times b_{i,ot}$
10. Calculate $\beta(t-1,i) = \beta(t-1,i) + \beta(t,j) \times p$
11. Calculate $\varepsilon(j,i) - \varepsilon(j,i) + \alpha(t-1,j) \times p \times \beta(t,i)/P(O\|\lambda)$
12. ENDFOR
13. ENDFOR
14. ENDFOR

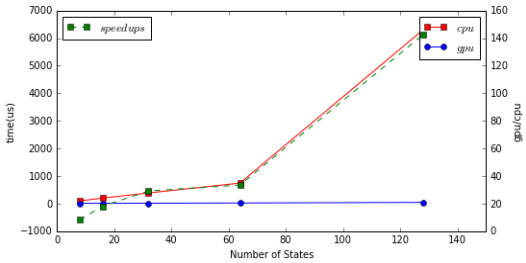
The following figures describes how the developed algorithm is able to track the real RUL even if perturbations occur. It shows the prognostic results of the engine RUL when fault occurs in the fan. In subplots (a) and (c), solid blue line denoted the estimated the health states sequence, meanwhile in subplots (b) and (d), solid blue line denoted the estimated RUL, dashed red line denoted real RUL. Simulation results show the algorithm robustness and its ability to predict RUL in case of faults. Simulations run only on CPU takes around 20mins, and only 8 seconds by using CPU and GPU.

V. EXPERIMENTAL RESULTS AND ANALYSIS

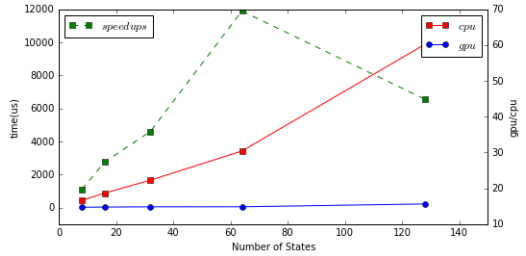
In this section data from CMAPSS simulator is used for algorithm testing. Data are referred to a turbofan engine simulates parameters values from 7 flight conditions. Testing environment is Intel Core i7-7700HQ CPU, 2.80GHz, 16GB RAM, GTX1050 Display Card, which includes 640 CUDA cores, i.e.10240 physical threads. The performance of algorithms used in GMM-HMM training and observation data implementation estimation procedures are shown in the Figure 2. Data pre-processing by GPU get around 850 speedups because all the data transformation is computed by GPU threads in parallel. Similarly, Monte-Carlo simulation get 210 times speedup when we adopt 5000 simulation runs in every RUL estimation. Similarly, we get a same speedup performance by running GMM cluster. Paper [8] gives Bayesian Information Criterion to determine the number of HMM states. In the case of this paper, the number of HMM states is 30. The speedups of Vertebi and Baum-Velch is around 35 and 25 respectively. Considering all implementation procedures in simulation, totally around 120 times speedups by using GPU parallel computing.



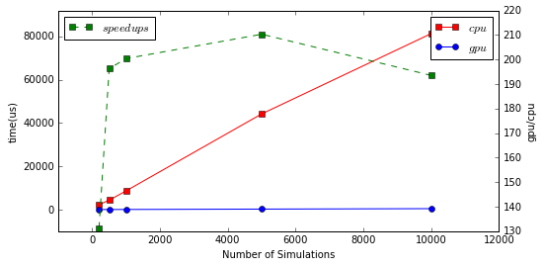
(a)Pre-Processing



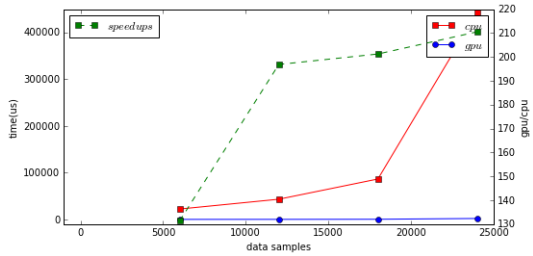
(b)Forward Algorithm



(c)Baum-Velch

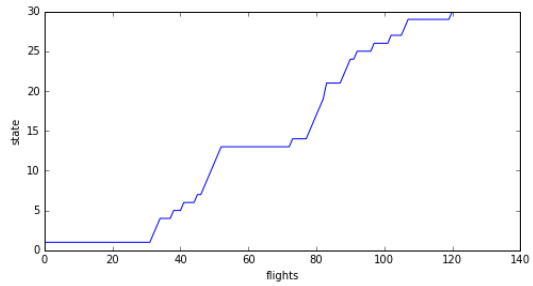


(d) Monte-Carlo

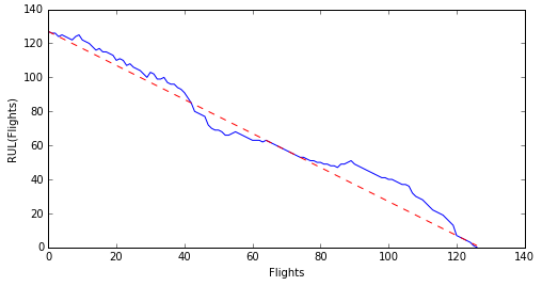


(e) Gaussian Mixture Model Cluster

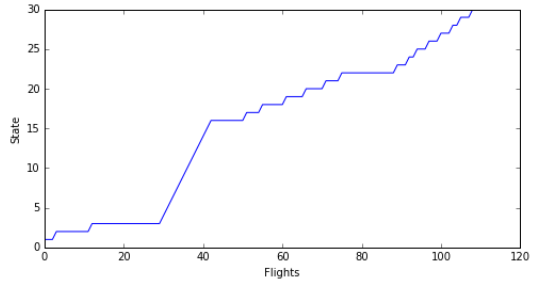
FIGURE II. PERFORMANCE OF ESTIMATION PROCEDURES ON CPU AND GPU



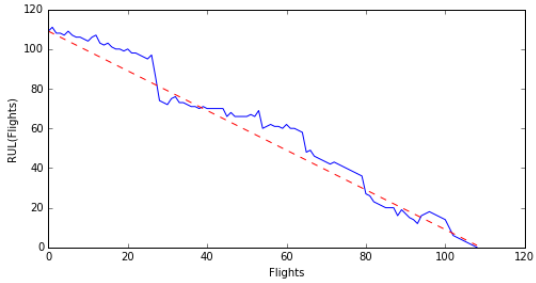
(a) Engine 1 Fan fault(health states)



(b) Engine 1 Fan fault(RUL estimation)



(c) Engine 1 Fan fault (health states)



(d) Engine 1 Fan fault (RUL estimation)

FIGURE III. HEALTH STATES SEQUENCE AND RUL ESTIMATION

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

VI. CONCLUSIONS

RUL estimation is becoming more and more interest topic because it promises to reduce spares inventory, maintenance costs and safety hazards.

In this paper an integrated GMM and HMM algorithm is developed for turbofan engines RUL estimation. Data from CMAPSS are used and simulations are performed to show effectiveness of this algorithm. It is obviously to find the accuracy of the prognostics method depends on the sample size used to construct the HMM, thus GPU parallel computing is necessary for a large amount of dataset. In this paper, simulation results show that parallel computing based on GPU can realize around 120 times speedup, however, our program is still not achieved the best performance of current hardware environment, optimization and different test on other hardware environment can be tested and developed in the future.

REFERENCES

[1] Haolei, Wu, Forecasting technology research for Aero-engine condition monitoring,[D],2009.

[2] Li,Y. Kurfess, T., and Liang, S.Y., 2000. "Stochastic prognostics for rolling element bearings". Mechanical Systems and Signal Processing, 14(5), September, pp.747-762.

[3] Goto, S., Afachi, Y., Katafuchi, S., Furue, T., Uchida, Y., Sueyoushi, M., Hatazazki, H., and Nakamura, M., 2008. "On-line deterioration prediction and residual life evaluation of rotating equipment based on vibration measurement". In Proceeding of the SICE conference, PP.812-817.

[4] Chunyan Ma, PHM Forecasting method research based on data mining technology. 2010[D]

[5] Yuwei Yang, Modern aero turbo fan engine fault analysis and intelligent diagnosis research.[D],2009.

[6] Wang,P., and Vachtsevanos, G., 2002. "Fault prognostics using dynamic wavelet neutal networks". Artificial Intelligence for Engineering Design, Analysis and Manufacturing. 15(11). January, pp.2085-2090

[7] Zhang, X., Xu, R., Kwan, C., Liang, S. Y., Xie, Q., and Haynes, L., 2005. "An integrated approach to bearing fault diagnostics and prognostics". In Proceedings of American Control Conference, pp. 2750-2755. [8] Thad Starner, Alex Pentland. Real-Time American Sign Language Visual Recognition From Video Using Hidden Markov Models. Master's Thesis, MIT, Feb 1995, Program in Media Arts

[8] Schwarz, G., 1978. "Estimating the dimension of a model". Annals of Statistics, 6(2), march, pp.461-464.