Розділ 4
Проектування інформаційних систем та програмних комплексів

UDC 004.451:004.054

SOFTWARE COMPLEX FOR INDEPENDENT STATE OF HEALTH (SOH) ASSESSMENT OF LAPTOP BATTERIES WITHOUT SPECIALIZED HARDWARE

OLEKSII BALENKO, MAKSYM GLAVCHEV, PAVLO NAUMENKO
(Oleksii.Balenko@khpi.edu.ua, Maksym.Glavchev@khpi.edu.ua,
Pavlo.S.Naumenko@cs.khpi.edu.ua)
National Technical University "Kharkiv Polytechnic Institute"

The paper presents the "BattDiag" software complex for independent State of Health (SoH) diagnostics of laptop batteries without the use of specialized hardware. Existing system utilities have an error margin of up to 10–15% as they merely relay data from the BMS controller. The proposed method is based on the mathematical analysis of real discharge curves using an improved Coulomb Counting algorithm with two-point drift correction. Experimental testing on real devices demonstrated that the developed complex provides a Mean Absolute Error (MAE) for SoH estimation of no more than 0.8%, which is 15 times more accurate than standard Windows tools, with a minimal background CPU load of 0.35%.

Under unstable power supply conditions, reliable assessment of the actual state of a laptop battery is a critical task. Standard Windows operating system tools (in particular, the powercfg /batteryreport report) and most third-party utilities do not perform independent measurements of battery parameters, but rather uncritically relay data from the BMS controller. Practical studies show that the error in estimating the State of Health (SoH) by such tools can reach 10–15% for highly degraded batteries. The aim of this work is to create an independent software method and a diagnostic complex based on the mathematical analysis of discharge curves.

Methodology and Architecture of the Complex. The proposed method is based on comparing the actual discharge curve in the "voltage–accumulated charge" space with a reference model. To implement the algorithm, the BattDiag software complex was created. It features a modular architecture and is implemented in Python using the psutil, NumPy, pandas, and tkinter libraries.

The primary method for calculating the actual capacity is Coulomb Counting with two-point drift correction. The calculation of the actual capacityті Q_{real} in a discrete implementation with a time step Δt is as follows:

$$Qreal = \Sigma^{N}_{i=1}\ I(t_i) \cdot \Delta t$$

where $I(t_i)$ is the discharge current value (A) at time t_i; $\Delta t = 30$ s is the time step; N is the number of measurements per cycle.

The SoH degradation coefficient (%) is calculated as:

$$SoHk = (Q_{real\_k} / Q_{design}) \times 100\%$$

where Q_{design} is the design capacity (mAh).

Research Results. To verify the developed software complex, a series of test experiments were conducted on three real laptops (A - new, B - medium wear, C - significant wear). All measurements were compared with a control discharge using a precision multimeter (Table 1)..

Table 1 - Results of SoH test measurements

Indicator	Laptop A	Laptop B	Laptop C
Q_{real} (BattDiag), мА·год	55 290	67 100	28 610
Q_{real} (еталон, Fluke), мА·год	55 505	67 640	28 220
SoH (BattDiag), %	97.0	78.0	59.6
SoH (еталон), %	97.4	78.7	58.8
SoH (Windows Battery Report), %	98.5	82.1	71.4
BattDiag Error (MAE SoH), %	0.4	0.7	0.8
Windows Error (MAE SoH), %	1.1	3.4	12.6
RMSE напруги (BattDiag vs еталон), В	0.018	0.045	0.089
Load class (Idle тест)	Idle ✓	Idle ✓	Idle ✓

Conclusions. The achieved accuracy of SoH estimation by the BattDiag software complex (MAE ≤ 0.8%) significantly exceeds the capabilities of built-in Windows tools, whose error reaches 12.6% (15 times higher). The CPU overhead of the complex during background logging is 0.35%, allowing it to be used regularly without impacting user performance..

REFERENCES

[1] Міністерство енергетики України, "Звіт про стан електроенергетичної галузі," Київ, 2024.

[2] G. L. Plett, *Battery Management Systems, Vol. I: Battery Modeling*. Norwood: Artech House, 2015.

[3] A. Gismero, E. Schaltz, and D.-I. Stroe, "Recursive State of Charge and State of Health Estimation Method for Lithium-Ion Batteries Based on Coulomb Counting and Open Circuit Voltage," *Energies*, vol. 13, no. 7, p. 1811, Apr. 2020, doi: 10.3390/en13071811.

[4] H. Huang *et al.*, "A comprehensively optimized lithium-ion battery state-of-health estimator based on Local Coulomb Counting Curve," *Appl. Energy*, vol. 322, p. 119469, Sep. 2022, doi: 10.1016/j.apenergy.2022.119469.

[5] Y. Zhou and B. Guo, "A Review on State of Health Estimation for Lithium-ion Batteries," *Int. J. Front. Eng. Technol.*, vol. 4, no. 10, pp. 14–21, 2022, doi: 10.25236/IJFET.2022.041003.

UDC 004.4

A GENERALIZED TRANSPILER ARCHITECTURE FOR LANGUAGES TARGETING GO

FORKERT P. P., IVANCHENKO M. G.
(fxposter@gmail.com, Sidorova.m.g@gmail.com)
Oles Honchar Dnipro National University

This work presents a generalized multi-stage transpiler architecture for implementing programming languages, dialects, and domain-specific languages that compile to Go through source-to-source translation. The architecture identifies nine sequential stages — from parsing through to delegation to the standard Go toolchain — and introduces normalization as a central mechanism that decouples language-specific constructs from Go code generation. The approach has been experimentally validated through the GoNext dialect, which implements eight distinct language extensions atop Go without modifying its official compiler.

Modern language implementation increasingly relies on reusing existing platforms [1, 2]. While JVM and LLVM have been extensively studied as host platforms, Go's suitability for this role remains underexplored despite its practical advantages: a fast compiler, an efficient runtime with a modern