Надёжность информационных систем. Лабораторная работа №3. Оценка безотказности нерезервированных систем: прикидочный, ориентировочный и окончательный расчёты

Макаров П. А.

Содержание

1 Введение **1**

2 Краткая теория **2**
2.1 Прикидочный расчёт 2
2.2 Ориентировочный расчёт 3
2.3 Окончательный расчёт 3
2.4 Проблема достоверности данных 3

3 Практическое задание **3**
3.1 Системный анализ и актуализация данных о поправочных коэффициентах 3
3.2 Аналитический расчёт 4
3.3 Статистическое моделирование на Python 4

4 Требования к отчёту **8**

5 Контрольные вопросы **8**

Список источников **9**

1 Введение

В Лекции 5 конспекта [1] рассматривается *методика оценки безотказности нерезервированных систем*. В частности, автор выделяет три уровня расчёта надёжности: **прикидочный**, **ориентировочный** и **окончательный**, — каждый из которых применяется на своём этапе проектирования ИС и отличается полнотой учитываемых факторов.

В то же время в тексте конспекта приводятся справочные данные о компонентах вычислительной техники, характерные для 1980 – 1990-х годов, а также таблицы поправочных коэффициентов, разработанные для аппаратуры общего применения. Эти данные в силу морального устаревания элементной базы делают необходимым *переосмысление* методики расчёта надёжности и безотказности применительно к ИС, используемым в настоящее время.

Современный серверный узел — это не просто сумма компонентов, а сложная электромеханическая система, работающая в конкретных условиях эксплуатации:

- в охлаждаемом машинном зале дата-центра ($t_0 = 18 - 27\,^\circ\mathrm{C}$);
- в составе периферийной вычислительной системы (или так называемой edge-платформы) на промышленном объекте ($t_0 \in [-40, +70]\,^\circ\mathrm{C}$, вибрации, пыль, перепады влажности);
- в бортовой вычислительной системе ($t_0 \in [-55, +125]\,^\circ\mathrm{C}$, дополнительно к предыдущим внешим факторам возможны перепады давления, радиация).

Интенсивность отказов одного и того же NVMe-накопителя в этих условиях может различаться на *порядки*, что делает прикидочный расчёт заведомо грубым, а окончательный — практически значимым.

Цель работы

1. Изучить три уровня расчёта безотказности нерезервированных систем: прикидочный, ориентировочный и окончательный.
2. Провести *верификацию и актуализацию* таблиц поправочных коэффициентов из Лекции 5 [1] применительно к современному серверному оборудованию.
3. Выполнить сравнительный расчёт надёжности типового серверного узла в различных условиях эксплуатации.
4. Разработать программу на Python для анализа влияния внешних факторов (температуры, электрической нагрузки, высоты установки) на вероятность безотказной работы ИС.

2 Краткая теория

Приближённые формулы для высоконадёжных систем ($\lambda_{\text{sys}}\, t \ll 1$):

$$P(t) \approx 1 - \lambda_{\text{sys}}\, t, \qquad Q(t) \approx \lambda_{\text{sys}}\, t, \qquad T_{\text{ср}} \approx \frac{1}{\lambda_{\text{sys}}}. \tag{1}$$

Согласно Лекции 5 [1], расчёт надёжности нерезервированной системы с основным (последовательным) соединением элементов основан на следующих положениях.

2.1 Прикидочный расчёт

Применяется на этапе технического задания и эскизного проектирования. Допущения:

- все элементы изделия равнонадёжны: $\lambda_i = \lambda_{\text{экв}}$;
- интенсивности отказов не зависят от времени: $\lambda_{\text{экв}} = \text{const}$;
- отказ любого элемента приводит к отказу всей системы.

В этом случае результирующая интенсивность отказов:

$$\lambda_{\text{sys}} = N \cdot \lambda_{\text{экв}}, \tag{2}$$

где N — общее число элементов.

2.2 Ориентировочный расчёт

Применяется после разработки электрических схем. Учитывает *типы* элементов и их количество, но предполагает номинальный режим работы:

$$\lambda_{\text{sys}} = \sum_{i=1}^{r} N_i \cdot \lambda_{0i}, \tag{3}$$

где N_i — число элементов i-го типа, λ_{0i} — номинальная интенсивность отказов.

2.3 Окончательный расчёт

Применяется на этапе технического проектирования после лабораторных испытаний макетов. Учитывает реальные режимы работы и внешние воздействия:

$$\lambda_i = \lambda_{0i} \cdot \alpha_i \cdot k_i, \tag{4}$$

где:

- $\alpha_i = f(t_0, K_{\text{н}})$ — поправочный коэффициент, учитывающий температуру окружающей среды t_0 и электрическую нагрузку $K_{\text{н}}$;
- $k_i = f(j, \varphi)$ — поправочный коэффициент, учитывающий механические (j) и климатические (φ) воздействия.

2.4 Проблема достоверности данных

Таблицы поправочных коэффициентов в Лекции 5 [1] разработаны для радиоэлектронной аппаратуры 1980-х годов. Для современных полупроводниковых компонентов (CPU, NVMe, DDR5) зависимости $\alpha_i(t_0, K_{\text{н}})$ опираются на соответствующие физико-технологические модели [2; 3]: например, интенсивность отказов SSD экспоненциально растёт с температурой по закону Аррениуса, а для CPU критичным фактором является не столько температура, сколько *электромиграция* в транзисторах при высоких $K_{\text{н}}$.

Студенту необходимо найти модели деградации (например, модель Аррениуса для SSD, модель Коффина — Мансона для термоциклирования и паяных соединений), стандартные для современных ускоренных испытаний (ALT — Accelerated Life Testing) электронной компонентной базы, и адаптировать их под цель и задачи данной лабораторной работы.

3 Практическое задание

Задание включает системный анализ, аналитический расчёт и компьютерное моделирование.

3.1 Системный анализ и актуализация данных о поправочных коэффициентах

Рассмотрим типовой серверный узел для обработки больших данных. Студенту необходимо:

1. Выделить *три сценария эксплуатации* (например: облачный ЦОД, edge-промышленный сервер, бортовой компьютер).

2. Для каждого сценария определить диапазоны внешних воздействий: температура, влажность, вибрация, высота над уровнем моря.

3. Найти в открытых источниках современные модели зависимости λ_i от температуры и электрической нагрузки для ключевых компонентов: CPU, DDR5 RAM, NVMe SSD, блоков питания (PSU — Power Supply Unit), вентиляторов.

4. Составить таблицу поправочных коэффициентов α_i и k_i для каждого компонента в каждом сценарии.

Пример для поиска: согласно модели Аррениуса, ускорение деградации SSD при повышении температуры описывается как

$$\alpha_{\text{SSD}}(t_0) = \exp\left[\frac{E_a}{k_B}\left(\frac{1}{T_{\text{ref}}} - \frac{1}{T_0}\right)\right], \tag{5}$$

где $E_a \approx 0.7$ эВ — энергия активации, k_B — постоянная Больцмана, $T_{\text{ref}} = 313$ K (40 °C), T_0 — рабочая температура.

3.2 Аналитический расчёт

Используя найденные данные, а также приведённые ниже прикидочные значения, выполнить расчёт надёжности серверного узла *тремя методами.*

- **Метод 1 (прикидочный).** Принять $\lambda_{\text{экв}} = 50 \cdot 10^{-9}$ 1/ч (50 FIT), $N = 2\,000$ (типовой сервер). Рассчитать λ_{sys}, $P(t)$ за 1 год, $T_{\text{ср}}$.

- **Метод 2 (ориентировочный).** Разбить сервер на типы компонентов (CPU × 2, DIMM × 16, NVMe × 4, PSU × 2, вентиляторы × 8 и т. д.), подставить λ_{0i} из современных спецификаций. Рассчитать те же показатели.

- **Метод 3 (окончательный).** Для сценария «edge-промышленный сервер» ($t_0 = 60\,^{\circ}$C, вибрация $k_1 = 1.4$, влажность $k_2 = 2.5$) рассчитать $\lambda_i = \lambda_{0i} \cdot \alpha_i \cdot k_i$ для каждого компонента и суммарную λ_{sys}.

По результатам выполненных расчётов составить итоговую таблицу.

Таблица 1: Сравнение методов расчёта надёжности

Показатель	**Прикидочный**	**Ориентировочный**	**Окончательный**
λ_{sys}, 1/ч	?	?	?
$P(8\,760$ ч$)$	?	?	?
$T_{\text{ср}}$, лет	?	?	?

Сделать вывод о том, во сколько раз окончательный расчёт отличается от прикидочного, и объяснить физический смысл расхождения.

3.3 Статистическое моделирование на Python

Необходимо написать скрипт на Python, который:

- реализует три модели расчёта λ_{sys} (прикидочная, ориентировочная, окончательная);

- строит графики зависимости $P(t)$ от времени для трёх методов на интервале $t \in [0,\, 5 \cdot 10^4]$ ч;
- строит *поверхностный график* (surface plot) зависимости $P(8\,760\text{ ч})$ от температуры $t_0 \in [20,\, 80]\,^\circ\text{C}$ и коэффициента нагрузки CPU $K_\text{н} \in [0.1,\, 1.0]$;
- сравнивает теоретические значения $P(t)$ с результатами метода Монте-Карло (моделирование 10^4 реализаций работы сервера).

Пример кода

```
import math
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# 1. Исходные данные
T_max = 5e4           # время моделирования, часы
t_obs = 8760          # время наблюдения (1 год), часы
N_sim = 10000         # число реализаций для метода Монте-Карло

# --- Прикидочный расчёт ---
lambda_eq = 50e-9     # эквивалентная интенсивность отказов, 1/ч (50 FIT)
N_eq = 2000           # число элементов в сервере

# --- Ориентировочный расчёт (типовой сервер) ---
# Тип компонента : (количество N_i, lambda_0i в 1/ч)
components = {
    'CPU'      : (2,  50e-9),
    'DIMM'     : (16, 20e-9),
    'NVMe'     : (4,  30e-9),
    'PSU'      : (2,  100e-9),
    'Fan'      : (8,  500e-9),
    'NIC'      : (2,  40e-9),  # Network Interface Card
    'Backplane': (1,  10e-9),  # Объединительная (задняя) плата
}

# --- Окончательный расчёт (edge-сценарий) ---
# Поправочные коэффициенты для edge-сервера:
#   t0 = 60 C, вибрация k1 = 1.4, влажность k2 = 2.5
t0_edge = 60          # C
k_vib   = 1.4
k_hum   = 2.5

# Энергия активации для модели Аррениуса (типично для SSD и паяных
↪  соединений)
E_a     = 0.7         # эВ
k_B     = 8.617e-5    # эВ/K
T_ref   = 313.0       # K (40 C - номинальная температура)

def arrhenius_alpha(t_celsius, E_a=E_a, T_ref=T_ref):
```

```
    """Поправочный коэффициент по модели Аррениуса."""
    T = t_celsius + 273.15
    return math.exp((E_a / k_B) * (1.0 / T_ref - 1.0 / T))

# Коэффициент электрической нагрузки CPU (упрощённая модель)
def load_factor(K_n):
    """Ускорение деградации CPU из-за электромиграции."""
    return 1.0 + 3.0 * K_n**2

# 2. Расчёт lambda_sys тремя методами
# Метод 1: прикидочный
lambda_sys_approx = N_eq * lambda_eq

# Метод 2: ориентировочный
lambda_sys_orient = sum(N_i * lam_i for N_i, lam_i in components.values())

# Метод 3: окончательный (edge, t0=60 C, K_n=0.8 для CPU)
K_n_cpu = 0.8
alpha_temp = arrhenius_alpha(t0_edge)
alpha_cpu  = arrhenius_alpha(t0_edge) * load_factor(K_n_cpu)

lambda_sys_final = 0.0
for name, (N_i, lam_0) in components.items():
    if name == 'CPU':
        alpha_i = alpha_cpu
    else:
        alpha_i = alpha_temp
    k_i = k_vib * k_hum
    lambda_sys_final += N_i * lam_0 * alpha_i * k_i

print(f"lambda_sys (прикидочный)     = {lambda_sys_approx:.3e} 1/ч")
print(f"lambda_sys (ориентировочный) = {lambda_sys_orient:.3e} 1/ч")
print(f"lambda_sys (окончательный)   = {lambda_sys_final:.3e} 1/ч")

# 3. График P(t) для трёх методов
t = np.linspace(0, T_max, 1000)
P_approx = np.exp(-lambda_sys_approx * t)
P_orient = np.exp(-lambda_sys_orient * t)
P_final  = np.exp(-lambda_sys_final  * t)

plt.figure(figsize=(10, 6))
plt.plot(t / 1e3, P_approx, 'b-',  label='Прикидочный')
plt.plot(t / 1e3, P_orient, 'g--', label='Ориентировочный')
plt.plot(t / 1e3, P_final,  'r-.', label='Окончательный (edge, 60°C)')
plt.axvline(t_obs / 1e3, color='k', ls=':', label=f't = {t_obs} ч (1 год)')
plt.xlabel('Время t, тыс. часов')
plt.ylabel('Вероятность безотказной работы P(t)')
plt.title('Сравнение трёх методов расчёта надёжности')
plt.legend()
plt.grid(True, alpha=0.5)
```

```
plt.tight_layout()
plt.savefig('lab03_Pt_comparison.png', dpi=200)
plt.show()

# 4. Surface plot: P(8760 ч) = f(t0, K_n)
t0_range = np.linspace(20, 80, 30)
Kn_range = np.linspace(0.1, 1.0, 30)
T0, KN = np.meshgrid(t0_range, Kn_range)
P_surface = np.zeros_like(T0)

for i in range(T0.shape[0]):
    for j in range(T0.shape[1]):
        lam = 0.0
        for name, (N_i, lam_0) in components.items():
            alpha_T = arrhenius_alpha(T0[i, j])
            if name == 'CPU':
                alpha_i = alpha_T * load_factor(KN[i, j])
            else:
                alpha_i = alpha_T
            k_i = k_vib * k_hum
            lam += N_i * lam_0 * alpha_i * k_i
        P_surface[i, j] = math.exp(-lam * t_obs)

fig = plt.figure(figsize=(12, 7))
ax = fig.add_subplot(111, projection='3d')
surf = ax.plot_surface(T0, KN, P_surface, cmap='viridis', alpha=0.9)
ax.set_xlabel('Температура t0, °C')
ax.set_ylabel('Нагрузка CPU K_n')
ax.set_zlabel('P(8760 ч)')
ax.set_title('Зависимость ВБР за 1 год от температуры и нагрузки')
fig.colorbar(surf, shrink=0.5, aspect=10)
plt.tight_layout()
plt.savefig('lab03_surface_plot.png', dpi=200)
plt.show()

# 5. Сравнение с методом Монте-Карло (для окончательного расчёта)
rng = np.random.default_rng(42)
ttf_samples = rng.exponential(scale=1.0 / lambda_sys_final, size=N_sim)
P_mc = np.mean(ttf_samples > t_obs)
P_theory = math.exp(-lambda_sys_final * t_obs)

print(f"\nМонте-Карло: P({t_obs} ч) = {P_mc:.5f}")
print(f"Теория:      P({t_obs} ч) = {P_theory:.5f}")
print(f"Расхождение: {abs(P_mc - P_theory) / P_theory * 100:.2f}%")

plt.figure(figsize=(9, 5))
plt.hist(ttf_samples / 1e3, bins=100, density=True, alpha=0.6,
         label='Монте-Карло (окончательный)')
x = np.linspace(0, max(ttf_samples) / 1e3, 500)
plt.plot(x, lambda_sys_final * np.exp(-lambda_sys_final * x * 1e3) * 1e3,
```

```
        'r-', lw=2, label='Теоретическая плотность')
plt.axvline(t_obs / 1e3, color='k', ls=':', label=f't = {t_obs} ч')
plt.xlabel('Время t, тыс. часов')
plt.ylabel('Плотность')
plt.title('Распределение наработки до отказа (окончательный расчёт)')
plt.legend()
plt.grid(True, alpha=0.5)
plt.tight_layout()
plt.savefig('lab03_monte_carlo.png', dpi=200)
plt.show()
```

4 Требования к отчёту

Отчёт должен содержать:

1. Титульный лист, цель работы и краткие теоретические положения.
2. Таблицу с актуализированными поправочными коэффициентами α_i, k_i для трёх сценариев эксплуатации (с указанием источников).
3. Результаты аналитических расчётов тремя методами с подробными выкладками и сравнительной таблицей.
4. Листинг программного кода с комментариями.
5. Графики, сгенерированные скриптом:
   - $P(t)$ для трёх методов расчёта;
   - surface plot $P(8\,760\text{ ч}) = f(t_0, K_\text{н})$;
   - сопоставление теоретической и статистической (Монте-Карло) оценок $P(t)$.
6. Анализ того, почему при высоких температурах и нагрузках приближённая формула (1) перестаёт работать и необходимо использовать точное выражение $P(t) = e^{-\lambda t}$.
7. Выводы о применимости каждого метода расчёта на различных этапах проектирования ИС.

5 Контрольные вопросы

1. Почему для высоконадёжных систем ($\lambda t \ll 1$) справедливо приближение $P(t) \approx 1 - \lambda t$?
2. В чём принципиальное отличие прикидочного, ориентировочного и окончательного расчётов надёжности? На каких этапах проектирования ИС применяется каждый из этих методов?
3. Почему в прикидочном расчёте допускается допущение $\lambda_i = \lambda_\text{экв}$, и в каких случаях оно приводит к грубым ошибкам?
4. Почему метод окончательного расчёта требует лабораторных испытаний макетов?
5. Какой физический смысл имеют поправочные коэффициенты α_i и k_i?

6. Как связаны между собой коэффициент электрической нагрузки $K_{\text{н}}$ и явление электромиграции в современных транзисторах?

7. Как температура окружающей среды влияет на интенсивность отказов NVMe SSD согласно модели Аррениуса?

8. Как изменится λ_{sys} серверного узла при переходе из охлаждаемого ЦОД ($t_0 = 22\,^\circ\text{C}$) в промышленный шкаф ($t_0 = 60\,^\circ\text{C}$)?

Список источников

Книги и учебные пособия

1. *Расулова С. С.* Надёжность информационных систем. — Ташкент : Ташкентский университет информационных технологий, 2007. — С. 118.
2. *Pecht M.* Product Reliability, Maintainability, and Supportability Handbook. — 2rd Edition. — Boca Raton : CRC Press, 2009. — P. 480. — DOI: 10.1201/9781420009897.
3. *Modarres M., Kaminskiy M. P., Krivtsov V.* Reliability Engineering and Risk Analysis: A Practical Guide. — 3rd Edition. — Boca Raton : CRC Press, 2016. — P. 522.

Актуальные отраслевые стандарты и документация

Отраслевые стандарты

- ГОСТ Р ИСО/МЭК 30134-2018: Информационные технологии. Центры обработки данных. Ключевые показатели эффективности.
- JEDEC (Joint Electron Device Engineering Council) JEP 122 — Failure Mechanisms and Models for Semiconductor Devices.
- JEDEC JESD218 — Solid State Drive (SSD) Requirements and Endurance Test Method
- Telcordia SR-332: Reliability Prediction Procedure.

Спецификации производителей

- Intel Xeon Scalable Processors: Reliability, Availability, and Serviceability.
- Micron 3D TLC NAND Flash Memory: Architecture, Generations, Endurance, Product Ecosystem and System Behavior.

Открытые исследования и отчёты ЦОД

- Backblaze Drive Stats Quarterly Reports.
- Google Datacenter Reports.