Тестове діагностування цифрових пристроїв

**Тестове діагностування** - це [технічне діагностування](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%85%D0%BD%D1%96%D1%87%D0%BD%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%80%D0%B0%D0%B4%D1%96%D0%BE%D0%B5%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2), при якому на об'єкт подаються спеціальні впливи, що використовуються тільки для цілей діагностування. Впливи, що подаються на об'єкт при тестовому діагностуванні, називаються **тестами діагностування** або просто **тестами** ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0" \o "Англійська мова) *tests*) .

**Зміст**

* [1Несправності та тести](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%9D%D0%B5%D1%81%D0%BF%D1%80%D0%B0%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%96_%D1%82%D0%B0_%D1%82%D0%B5%D1%81%D1%82%D0%B8)
* [2Загальна класифікація методів побудови тестів](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%97%D0%B0%D0%B3%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%96%D0%B2_%D0%BF%D0%BE%D0%B1%D1%83%D0%B4%D0%BE%D0%B2%D0%B8_%D1%82%D0%B5%D1%81%D1%82%D1%96%D0%B2)
* [3Алгоритмічні тести](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%96%D1%87%D0%BD%D1%96_%D1%82%D0%B5%D1%81%D1%82%D0%B8)
* [4Псевдовипадкові методи побудови тестів](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%9F%D1%81%D0%B5%D0%B2%D0%B4%D0%BE%D0%B2%D0%B8%D0%BF%D0%B0%D0%B4%D0%BA%D0%BE%D0%B2%D1%96_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8_%D0%BF%D0%BE%D0%B1%D1%83%D0%B4%D0%BE%D0%B2%D0%B8_%D1%82%D0%B5%D1%81%D1%82%D1%96%D0%B2)
* [5Детерміновані методи побудови тестів](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%94%D0%B5%D1%82%D0%B5%D1%80%D0%BC%D1%96%D0%BD%D0%BE%D0%B2%D0%B0%D0%BD%D1%96_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8_%D0%BF%D0%BE%D0%B1%D1%83%D0%B4%D0%BE%D0%B2%D0%B8_%D1%82%D0%B5%D1%81%D1%82%D1%96%D0%B2)
* [6Джерела](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D0%B5_%D0%B4%D1%96%D0%B0%D0%B3%D0%BD%D0%BE%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D1%85_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D0%BE%D1%97%D0%B2#%D0%94%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0)

Несправності та тести

[Цифрові схеми](https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D0%B2%D0%B5%D0%BD%D1%82%D0%B8%D0%BB%D1%8C), як і будь-які фізичні пристрої, в процесі своєї роботи можуть мати порушення працездатності. Причиною порушення працездатності є *дефекти*. Під **[дефектом](https://uk.wikipedia.org/wiki/%D0%94%D0%B5%D1%84%D0%B5%D0%BA%D1%82" \o "Дефект)** в технічній діагностиці розуміється фізичне пошкодження, що призводить до відмови в роботі пристрою тобто до стабільної втрати працездатності. Математичною моделлю дефекту є [**несправність**](https://uk.wikipedia.org/wiki/%D0%9D%D0%B5%D1%81%D0%BF%D1%80%D0%B0%D0%B2%D0%BD%D1%96%D1%81%D1%82%D1%8C).

Серед усієї безлічі класів несправностей, існуючих в цифрових пристроях, можна виділити клас *константних* несправностей. Під **константною несправністю** розуміється постійне значення «логічний 0» або «логічна 1» на лінії цифрової схеми. Якщо в схемі виникає одна константна несправність, вона називається **одиночною константною несправністю (ОКН)**. Якщо в схемі одночасно присутні кілька ОКН, то така несправність називається **кратною**. Кількість ОКН схеми дорівнює подвоєному числу її ліній (за рахунок двох типів несправностей). Кількість кратних несправностей кратності � для схеми, що має � ліній, обчислюється за формулою:

�=�=1����∗2�=1

*Наприклад, для елемента 3-І, схема якого показана на рисунку 1, загальне число кратних несправностей буде 80.*

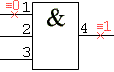
[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:%D0%95%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82_3-%D0%86_%D0%B7_%D0%BD%D0%B5%D1%81%D0%BF%D1%80%D0%B0%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%8F%D0%BC%D0%B8.gif)

Рисунок 1

На схемах константні несправності позначаються «хрестиком» на відповідній лінії із зазначенням типу несправності: ≡0 (константа 0) і ≡1 (константа 1).

У тексті константні несправності позначаються наступним чином : 10 - константна несправність нуля на першій лінії, 41 - константна несправність одиниці на п'ятій лінії.

Даному класу несправностей відповідають у реальності такі фізичні дефекти: обрив в ланцюзі [емітера](https://uk.wikipedia.org/wiki/%D0%95%D0%BC%D1%96%D1%82%D0%B5%D1%80), бази, колектора; короткі замикання емітер-база, база-колектор; обрив в ланцюгах резисторів. Іншими логічними несправностями можуть бути короткі замикання контактів, їх переплутування при монтажі схеми, зміни функцій окремих елементів або поєднання згаданих випадків.

*Тут і далі розглядаються константні несправності, якщо не зазначено інше.*

Якщо несправність змінює значення сигналу на виході схеми хоча б на одному з вхідних наборів, така несправність називається **суттєвою**, а якщо ні - **несуттєвою**.

Несуттєві несправності зазвичай пов'язані з надлишковими схемними структурами. На рисунку 2 наведена схема, у якій вхід X1 є надлишковим, а будь-яка несправність на ньому - несуттєвою.

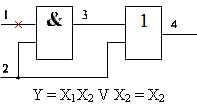
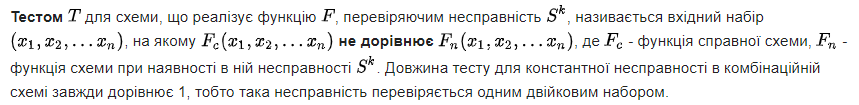
[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B7_%D0%BD%D0%B0%D0%B4%D0%BB%D0%B8%D1%88%D0%BA%D0%BE%D0%B2%D0%B8%D0%BC_%D0%B2%D1%85%D0%BE%D0%B4%D0%BE%D0%BC_X1.gif)

Рисунок 2

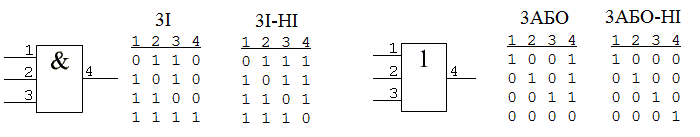
Якщо різні несправності при подачі на схему вхідних наборів дають різні реакції хоча б на одному наборі, такі несправності називаються **розрізнюваними**, а якщо ні - то **нерозрізнюваними**. Нерозрізнювані несправності утворюють **клас еквівалентних дефектів (КЕД)**. Прикладом нерозрізнюваних дефектів можуть бути несправності 10 і 40 для трьохвходового елемента І.



Довжина тесту для константної несправності в комбінаційній схемі завжди дорівнює 1, тобто така несправність перевіряється одним двійковим набором.

Будь-який [цифровий пристрій](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D1%80%D1%96%D0%B9) може мати один справний стан і безліч несправних (за кількістю розглянутих несправностей). Тест, що розрізняє справний стан схеми від безлічі несправних станів, називається **перевіряючим тестом**, а тест, що розрізняє несправні стани між собою називається **тестом пошуку дефекту**.

Сукупність тестових наборів, які перевіряють всі константі несправності схеми називається **тестом перевірки справності**. Для елементів логічного базису (І, АБО) тести перевірки справності мають довжину (�+1)- кількість входів елемента. На рисунку 3 наведено тести перевірки справності трьохвходових елементів І, І-НІ, АБО, АБО-НІ.

[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:%D0%A1%D1%85%D0%B5%D0%BC%D0%B8_%D1%82%D0%B5%D1%81%D1%82%D1%96%D0%B2_%D0%BF%D0%B5%D1%80%D0%B5%D0%B2%D1%96%D1%80%D0%BA%D0%B8_%D1%82%D1%80%D1%8C%D0%BE%D1%85%D0%B2%D1%85%D0%BE%D0%B4%D0%BE%D0%B2%D0%B8%D1%85_%D0%B5%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%96%D0%B2.gif.png)

Довжина тесту визначається кількістю двійкових наборів, що містяться у ньому. Повнота тесту виражається у *відсотках* (%) перевірюваних несправностей щодо загальної кількості несправностей схеми.

Загальна класифікація методів побудови тестів

Генерація тестів в загальному випадку складається з двох етапів: на першому етапі різними способами отримують вхідні тестові набори («кандидати у тест»), які потім аналізуються за спеціальними критеріями (моделюються), і ті набори, що задовольняють заданим умовам, включаються у тест.

За способом отримання вхідних тестових наборів («кандидатів у тест») методи поділяються на:

* **алгоритмічні методи генерації (генератори) тестів** : «кандидати в тест» генеруються апаратним чином;
* **алгоритми випадкового пошуку** : «кандидати в тест» генеруються випадковим чином;
* **спрямовані (детерміновані) алгоритми** : «кандидати в тест» генеруються для заданої несправності або зовнішнього входу схеми.

Алгоритмічні тести

**Алгоритмічні тести** - це певні види двійкових послідовностей, що генеруються апаратно з використанням спеціальних схем. Тип апаратного тесту залежить тільки від кількості входів пристрою, що перевіряється. Нижче, в таблиці 1 наведена структура вхідних впливів для різних видів n-розрядних алгоритмічних тестів.

Таблиця 1 - Основні типи алгоритмічних тестів.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вичерпний тест** | **0, що біжить** | **1, що біжить** | **Сусідній код** | **Шаховий код** |
| 1 2 3 ... n-1 n | 1 2 3 4 ... n | 1 2 3 4 ... n | 1 2 3 4 ... n | 1 2 3 4 ... n |
|  |  |  |  |  |
| 0 0 0 ... 0 0 | 0 1 1 1 ... 1 | 1 0 0 0 ... 0 | 1 0 0 0 ... 0 | 1 0 1 0 ... 0 |
| 0 0 0 ... 0 1 | 1 0 1 1 ... 1 | 0 1 0 0 ... 0 | 1 1 0 0 ... 0 | 0 1 0 1 ... 1 |
| 0 0 0 ... 1 0 | 1 1 0 1 ... 1 | 0 0 1 0 ... 0 | 1 1 1 0 ... 0 | 1 0 1 0 ... 0 |
| 0 0 0 ... 1 1 | 1 1 1 0 ... 1 | 0 0 0 1 ... 0 | 1 1 1 1 ... 0 | 0 1 0 1 ... 1 |
| ... ... ... ... ... | ... ... ... ... | ... ... ... ... | ... ... ... ... | ... ... ... ... |
| 1 1 1 ... 1 1 | 1 1 1 1 ... 0 | 0 0 0 0 ... 1 | 1 1 1 1 ... 1 | ... ... ... ... |

Способи апаратної генерації:

* «**Вичерпний тест**» - це 2� наборів, де � - число входів досліджуваної схеми. Спосіб генерації підсумовування (�=�+1) на �-розрядному лічильнику.
* «**Сусідній код**» - арифметичний зсув вправо послідовності 1 0 0 0 ... 0 ;
* «**0, що біжить**», «**1, що біжить**», «**Шаховий код**» - циклічний зсув вправо відповідає початковій послідовності .

Всі апаратні тести характеризуються високою швидкодією при одержанні «кандидатів у тест». Основний недолік - неможливість налаштування на конкретну перевірювану схему.

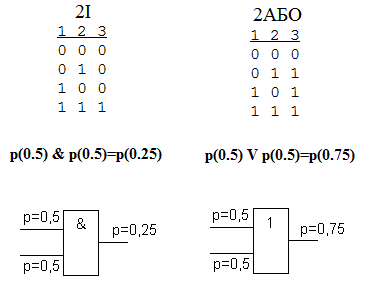
Псевдовипадкові методи побудови тестів

В алгоритмах випадкового пошуку «кандидати у тест» генеруються різного виду датчиками випадкових чисел. Зазвичай при випадковому пошуку «кандидат у тест» складається з одного набору і заздалегідь не орієнтований на перевірку будь-якої певної несправності.

Основні характеристики:

* Розміщення корисних вхідних наборів нерівномірно;
* Можливість управління щільністю ймовірності появи 0 і 1 у векторі;

При управлінні щільністю ймовірності появи «1» вираз*�Р* (%) вказує ймовірність появи певної % «1» в двійковій послідовності.

[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:%D0%99%D0%BC%D0%BE%D0%B2%D1%96%D1%80%D0%BD%D0%BE%D1%81%D1%82%D1%96_%D0%B5%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%96%D0%B2_2%D0%86_%D1%82%D0%B0_2%D0%90%D0%91%D0%9E.png)

Для включення «кандидата у тест», отриманого випадковим методом, як правило, використовується процедура **позитивного приросту**: кожен кандидат у тест моделюється «несправно» (обчислюється список виявляються даними набором несправностей), і, якщо сумарний список виявлених несправностей збільшується з урахуванням даного набору, то він включається в тест.

Процедура позитивного приросту:

1. Ймовірнісним методом генерується черговий набір «кандидата у тест».

2. Моделюються несправності, що перевіряються «кандидатом в тести».

3. «Кандидат у тест» включається в тест, якщо він збільшує % перевірюваних несправностей тесту.

Властивість імовірнісного генератора тестів - **режим «насичення»** при досягненні повноти тесту приблизно 90-92% . **Режим «насичення»** - це відсутність збільшення % перевірених несправностей з плином часу.

Алгоритми випадкового пошуку відносно прості в реалізації, мають досить високу швидкодію і дозволяють отримувати тести з досить високим відсотком перевірки несправностей для комбінаційних схем і ряду класів схем з пам'яттю. Все це зумовило їх широке застосування в системах побудови тестів. Ці алгоритми різняться способами формування випадкових вхідних наборів, критеріями включення"кандидатів" в тест, критеріями закінчення роботи, способами взаємодії з моделюванням тощо.

У найпростішому випадку за допомогою датчика випадкових генерується булев вектор, кожен розряд якого з рівною ймовірністю приймає значення 0 або 1. Цей вектор і слугує черговим вхідним набором – «кандидатом на тест». Для підвищення ефективності випадкового пошуку розроблений ряд евристичних алгоритмів генерації вхідних наборів. Всі ці алгоритми неявно засновані на припущенні, що «корисні» набори (набори, гідні включення в тест) у просторі вхідних наборів розміщуються нерівномірно.

Отже, для кожної схеми є області з відносно великим числом «корисних» наборів. Для того щоб виявити і використовувати такі області, або попередньо аналізують структуру схеми, або застосовують адаптивні алгоритми управління датчиком випадкових чисел. Наприклад, відомі алгоритми випадкового пошуку, в яких використовуються датчики, які формують випадкові вхідні набори з заданим числом одиниць �. Спочатку інтервал [�,�] можливих значень � розбивається на кілька рівних частин, для кожної з яких формується ряд пробних «кандидатів в тести» і оцінюються їх перевіряючі властивості. За результатами цих проб визначаються оптимальні області значень �, у яких надалі і проводиться формування вхідних наборів.

Детерміновані методи побудови тестів

Спрямовані алгоритми, навпаки, формують «кандидати в тести», призначені для перевірки заданої несправності або заданого ділянки шляху в схемі. Використовувані алгоритми, як правило, являють собою розвиток відповідних алгоритмів для комбінаційних схем. Розглянемо один з найпоширеніших алгоритмів - евристичний алгоритм активізації шляхів, заснований на D-алгоритмі для комбінаційних схем. Всі спрямовані алгоритми пов'язані з перебором варіантів, іноді надзвичайно великим, тому час пошуку кандидатів в тести» для заданої несправності зазвичай обмежується програмно.

**Детерміноване побудова тестів** ==> NP-повна задача

Завдання побудови тесту :

* вибір ланцюжка елементів (шляху до зовнішнього виходу);
* забезпечення умов транспортування несправності на вихід;
* довизначення умов транспортування на зовнішніх входах.

Процедури імплікації:

**пряма імплікація** - обчислення вихідних значень за вхідними;

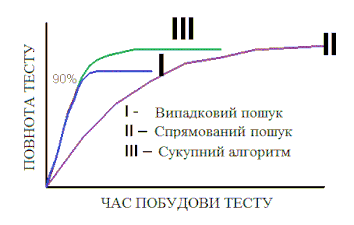
**зворотна імплікація** - обчислення значень по вихідних.

**Методи**:

* метод активізації шляхів;
* метод еквівалентних нормальних форм;
* метод булевих різниць;
* метод активізації шляхів (D-алгоритм і його модифікації).

Порівняємо [ефективність алгоритмів](https://uk.wikipedia.org/wiki/%D0%95%D1%84%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D1%96%D1%81%D1%82%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%83) довільного і спрямованого пошуку. При випадковому пошуку спочатку за короткий час перевіряється значна частина несправностей (тих, для яких число тестів досить велике), а потім ефективність алгоритму різко знижується (залишаються несправності, які перевіряються малим числом тестів, тому ймовірність випадкової генерації цих тестів мала). Це ілюструється кривою  � (графік має якісний характер).

Зазначимо, що час випадкової генерації � набагато менше часу моделювання. Для спрямованих алгоритмів час побудови порівнянний з часом моделювання, а в ряді випадків значно більше його. Спрямовані алгоритми, таким чином, виконуються повільніше алгоритмів випадкового пошуку, але забезпечують велику повноту тестів (крива ��). Найбільш доцільно використовувати алгоритми: спочатку алгоритм випадкового пошуку. А після зниження його ефективності до певного рівня спрямований алгоритм (крива ���).

[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:%D0%95%D1%84%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D1%96%D1%81%D1%82%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%96%D0%B2_%D0%B4%D0%BE%D0%B2%D1%96%D0%BB%D1%8C%D0%BD%D0%BE%D0%B3%D0%BE_%D1%96_%D1%81%D0%BF%D1%80%D1%8F%D0%BC%D0%BE%D0%B2%D0%B0%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BF%D0%BE%D1%88%D1%83%D0%BA%D1%83.gif)

Якщо випадковий пошук служить єдиним алгоритмом побудови тестів в системі діагностичного матзабезпечення, то його робота зазвичай закінчується після закінчення заданого часу. При спільному використанні випадкового пошуку і спрямованого алгоритму випадковий пошук закінчується, коли його ефективність стає нижче передбачуваної ефективності спрямованого алгоритму. Для цього в процесі випадкового пошуку значення деякого параметра, що характеризує ефективність алгоритму, періодично порівнюється із заздалегідь заданим значенням. Такими параметрами можуть бути час безрезультатного пошуку, середній час перевірки однієї несправності, кількість відкинутих «кандидатів в тести» на один прийнятий, число виходів з «глухого кута» і т. п.