

ТЕСТВАНЕ И ДИАГНОСТИКА

Упражнение 5

Автоматично генериране на тестови вектори

PODEM - алгоритъм

Подобрява D-алгоритъма. Базира се на симулиране на работата на схемата. Във възлите на търсещия граф се разполагат само първични входове. Използва оценките за контролируемостта на връзките.

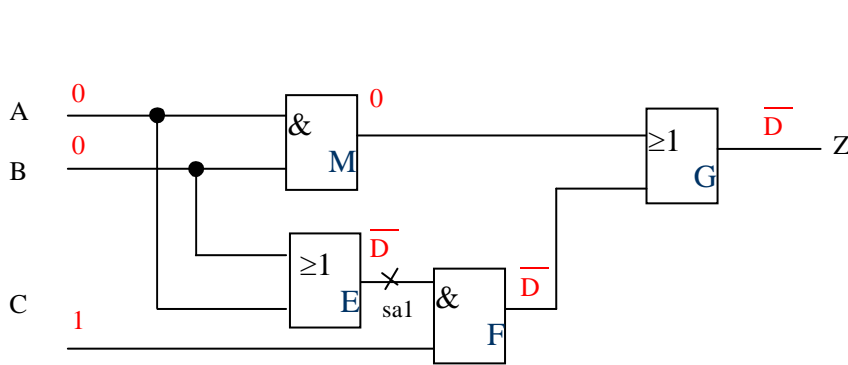
Алгоритъм:

1. Прави се право обхождане от повредената връзка за да се намери най-близкия първичен изход. Критерий е броят на комбинационните елементи, през които трябва да се премине за да се достигне до съответния изход.
2. Прави се обратно обхождане от повредената връзка, като се трасира пътя с най-трудна контролируемост.
3. На открития първичен вход се задава 0(1).
4. Симулира се поведението на схемата, като се контролира състоянието на повредената връзка и наблюдавания първичен изход.
 - a. Ако се регистрира конфликт (повредената връзка или първичният изход е в състояние 0 или 1) и на първичния вход е подадена 0(1), на същия първичен вход се подава алтернативната стойност на сигнала - 1(0). Изпълнява се т.4.
 - b. Ако се регистрира конфликт и на първичния вход вече е подадена алтернативната стойност, се връщаме към предходния първичен вход и му подаваме алтернативна стойност. Изпълнява се т.4.
 - c. Ако се регистрира конфликт и на всички изследвани първични входове е подадена алтернативната стойност следва, че не може да се намери тестващ вектор за маркираната повреда и работата се прекратява.
 - d. Ако повредената връзка и първичният изход са в състояние X, се прави ново обратно обхождане, за да се открие следващия първичен вход, който влияе на състоянието на повредената връзка и се изпълнява отново т.4.
 - e. Ако повредената връзка е в състояние D, а първичният изход е в състояние X, се изпълнява т.5.
 - f. Ако повредената връзка и първичният изход са в състояние D или \bar{D} следва, че тестващият вектор е открит и работата се прекратява.
5. Продължава се по пътя на разпространение на повредата към най-близкия първичен изход. За всеки комбинационен елемент по пътя се прави обратно обхождане (по пътя на най-трудно контролируемата връзка) и по аналогия на т.4 се изследва поведението на схемата.
6. Работата приключва, когато се намери входен вектор, който позволява повредата да се прояви на избрания първичен изход.

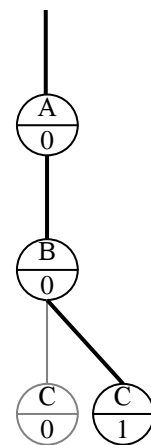


Пример 1 Като използвате алгоритъма Podem изградете търсещ граф и намерете тестващ вектор за повреда от тип sa1 на изхода на елемент E от схемата на фиг.1.

Проследяваме разпространението на повредата. Най-близкия първичен изход (в случая и единствен) е Z. Двата входа на елемент E са с еднаква контролируемост. Задаваме стойност 0



фиг.1.d

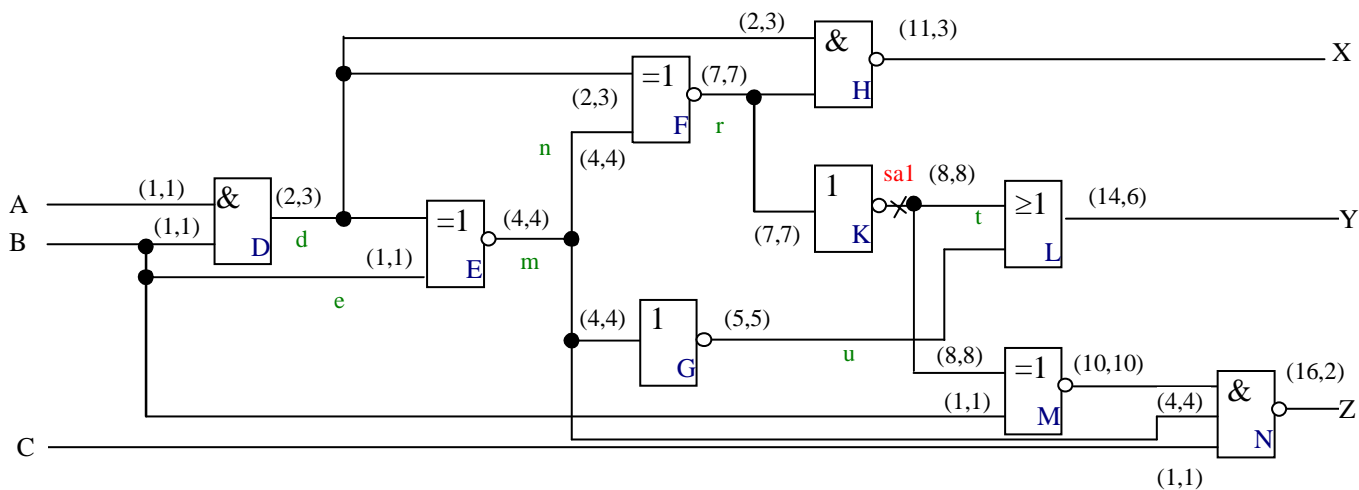


фиг.2.d

Продължаваме по пътя на разпространение на повредата. Състоянието на изхода на елемент F зависи от вход C. Задаваме стойност 0 на вход C (фиг.1.c, фиг.2.c). Това привежда изхода на F в състояние 0. Следователно достигнахме до конфликт. Задаваме стойност 1 на вход C (фиг.1.d, фиг.2.d). Сега на изхода на F излиза повреден сигнал \bar{D} . Изхода на елемент G (респективно първичния изход Z) е в състояние \bar{D} и за тестващ вектор получихме (001).



Пример 2 Като използвате алгоритъма Podem изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa0 на изхода на елемент K от схемата на фиг.3.

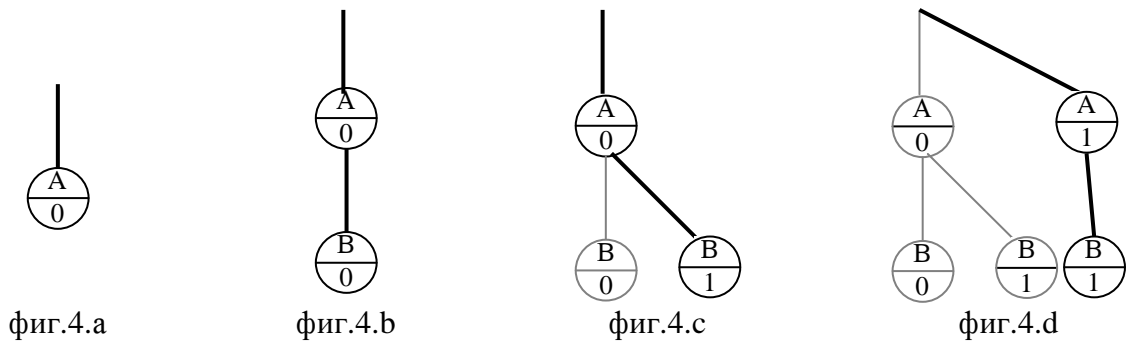


Фиг.3

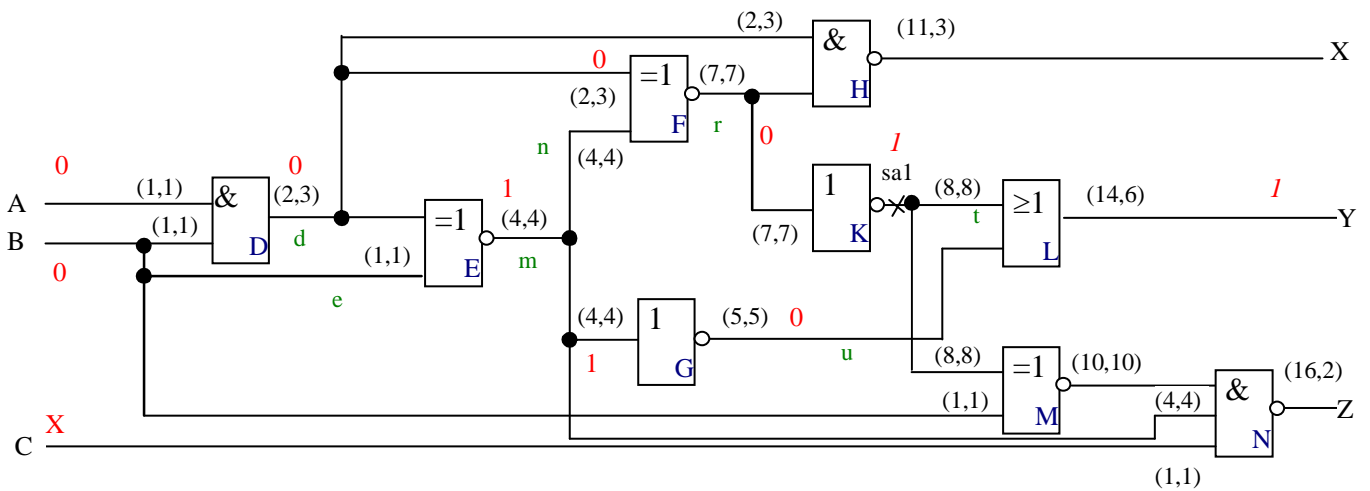
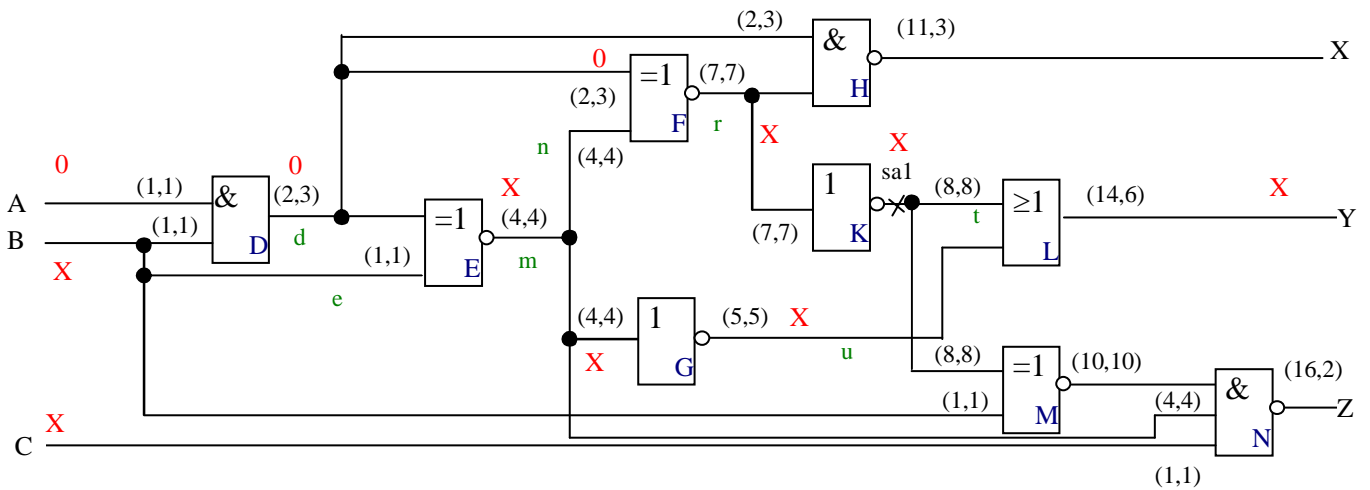
Проследяваме разпространението на повредата. Y се намира след един комбинационен елемент, а Z – след 2. Следователно най-близкия първичен изход е Y.

За да можем да доведем изхода на елемент K в състояние \bar{D} е необходимо изходът на елемент F да е в състояние 1. Елемент F има 2 входа, от които вход p е с по-високи стойности на контролируемостта. Сигнала за входа p се взема от изхода на елемент E, който от своя страна има също 2 входа. Входа d е с по-високи стойности на контролируемостта. Сигнала за входа d се взема от изхода на елемент D. Двата входа на елемент D се явяват първични входове (с еднаква контролируемост).

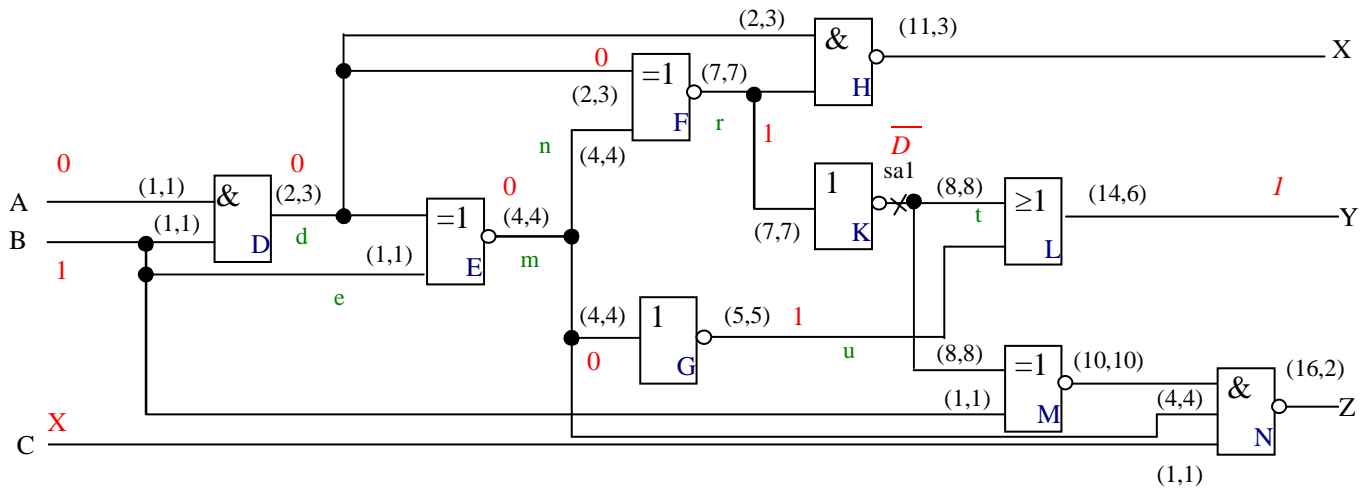
Задаваме стойност 0 на вход A(фиг.4.a) и симулираме поведението на схемата. Това привежда изхода на D в състояние 0, изхода на E в състояние X и изхода на F в състояние X(фиг.3.a).



Задаваме 0 и на вход B(фиг.4.b). Симулираме поведението на схемата (фиг.3.b). Изходът на елемент F е в състояние 0 и следователно изхода на K е стабилно в състояние в 1 (вместо желаното \bar{D}). Получава се конфликт.

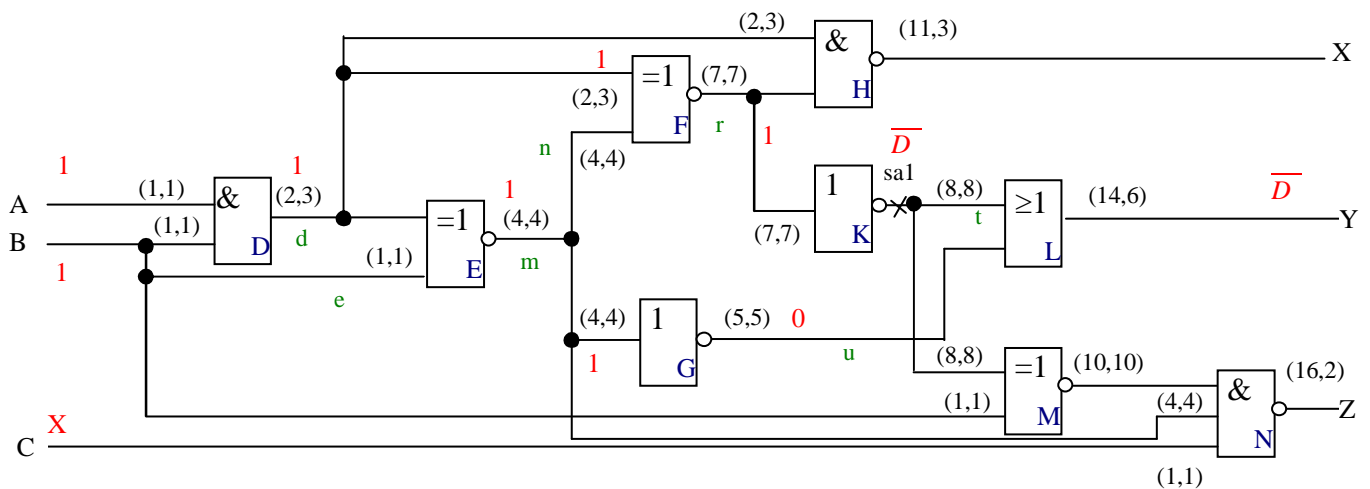


Променяме състоянието на входа В в 1(фиг.4.с). Симулираме поведението на схемата(фиг.3.с). Изходът на елемент К вече е в състояние \bar{D} , но изходът на L е стабилно в състояние 1. Отново имаме конфликт.



Фиг.3.с

Тъй като сме изчерпали всички възможни състояния за възел В, се връщаме един възел назад в търсеция граф и променяме състоянието на възел А в 1(фиг.4.d). Симулираме поведението на схемата(фиг.3.d). Изходът на елемент К е в състояние \bar{D} и изходът на L вече е в състояние \bar{D} . Няма конфликт. Векторът, който открива маркираната повреда е (11X).



Фиг.3.d

FAN - алгоритъм

Подобрява Podem-алгоритъма. Стреми се да намали работата при обратното обхождане. Да не се достига винаги до първичен вход, а до елемент, на базата на състоянието на който лесно и без опасност от промяна на състоянието на връзките в схемата, могат да се определят състоянията на първичните входове, които го предшестват.

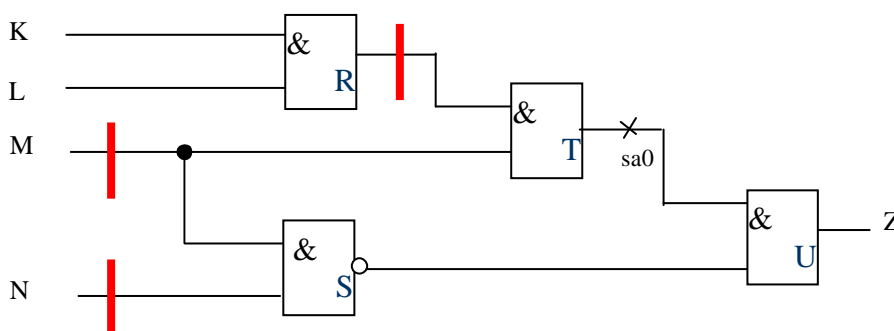
Термини:

- **свободни линии** – изходи на комбинационни елементи, чиито предшественици не участват в никакви възли;
- **главни линии (headlines)** – свободни линии, които се явяват входове на комбинационни елементи, които са част от накакво разклонение;
- **клони (fanout stem)** – изходите на разклоненията.

Възлите в графа могат да се разделят да 2 типа: клони и главни линии. Например възлите K, L, R, M и N от фиг.5 са свободни линии, но само R, M и N са главни линии. K и L се явяват предшественици на R. Обратното обхождане спира когато се достигне до главна линия.

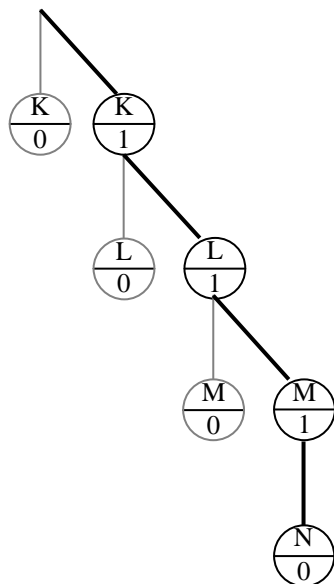


Пример 3 Като използвате алгоритъма FAN изградете търсещ граф и намерете тестваш вектор за откриване на повреда от тип sa0 на изхода на елемент T от схемата от фиг.5.

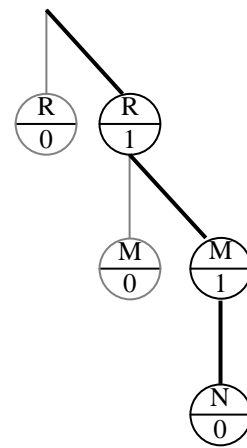


фиг.5

На фиг.5 с червени линии са показани главните линии за схемата. На фиг.6.a е показан търсещия граф за маркираната повреда според алгоритъма PODEM, а на фиг. 6.b – според алгоритъма FAN.



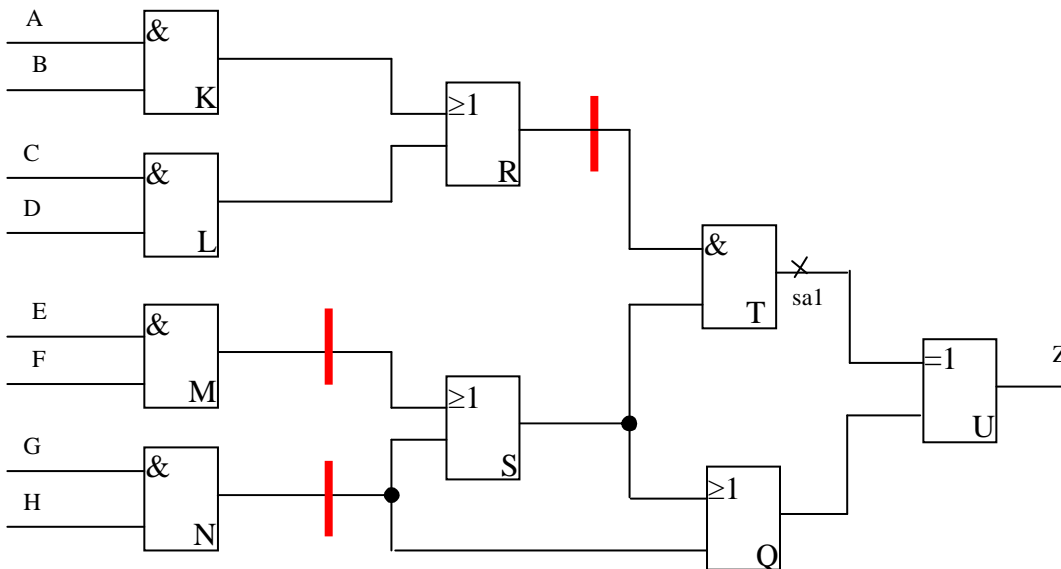
фиг.6.a



фиг.6.b



Пример 4 Като използвате алгоритъма FAN изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa1 на изхода на елемент T от схемата от фиг.7.



фиг.7



фиг.8

Свободните линии са: A, B, C, D, E, F, G, H, K, L, M, N и R. Главните линии са: R, M и N. На фиг.8 е показан търсеция граф за маркираната повреда съгласно алгоритъма FAN.



Задача 1. Като използвате Podem алгоритъм изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa0 на изхода на елемент G от схемата от фиг.3.



Задача 2. Като използвате Podem алгоритъм изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa1 на изхода на елемент F от схемата от фиг.3.



Задача 3. Като използвате Podem алгоритъм изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa0 на изхода на елемент E от схемата от фиг.3.



Задача 4. Изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa0 на изхода на елемент R от схемата от фиг.3. Решете задачата в 2 варианта- веднъж с използване на Podem алгоритъм и втори път с използване на Fan алгоритъм.



Задача 5. Изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa0 на изхода на елемент S от схемата от фиг.3. Решете задачата в 2 варианта- веднъж с използване на Podem алгоритъм и втори път с използване на Fan алгоритъм.



Задача 6. Изградете търсещ граф и намерете тестващ вектор за откриване на повреда от тип sa1 на изхода на елемент N от схемата от фиг.3. Решете задачата в 2 варианта- веднъж с използване на Podem алгоритъм и втори път с използване на Fan алгоритъм.