En studie i röj

av

Petter Lindgren

Examensarbete i matematisk statistik
Umeå universitet, 2000

Handledare: Professor Lennart Bondesson

Abstract

This paper presents a study of the computer game ”Minesweeper”.

The aim of the game is to search through a rectangular area of mined squares without hitting any mines. By using a strategy based on making every operation as safe as possible, series of the game have been simulated. The size of the playground is four times four squares. The simulations indicate how often the game will succeed and which square is the best one to start at. The strategy demands advanced mathematical calculations. The account of these is the major part of my work. My investigation shows that if there are three hidden mines the game will succeed about two times out of three. It also shows that the best startingpoint is a corner.

Keywords: Binary variables, Gauss Jordan elimination, independent island, neighbour, probability, simulation, system of equations.

Acknowledgements

I would like to thank my supervisor, Professor Lennart Bondesson for introducing me to the problem and for valuable help during my work. I would also like to thank Peter Anton and Lennart Nilsson for always supporting me. At last, I would like to say that the atmosphere at the Department of Mathematical Statistics at University of Umeå created by a very helpful staff has been a great help to me. Thank you all.

Innehållsförteckning

1. Introduktion *1*

2 Några exempel *2*

3 Matematisk teori *9*

4 Från teori till simulering *19*

4.1 Utökad grannmatris **19**

4.2 Minutlottning **20**

4.3 Hur spelet börjar **20**

4.4 Spelet fortsätter **21**

4.5 En simuleringsstudie **22**

5 Simuleringsresultat *25*

5.1 Intervallskattning **26**

5.2 Hypotesprövning **28**

6 Diskussion *30*

Referenser *32*

Appendix A Algoritm, spela ett parti Ms röj ..... *I*

Appendix B Källkod ..... *II*

1. Introduktion

MS röj, på engelska Minesweeper, är ett strategispel som följer med i standardversionen av Windows.

Spelplanen består av ett rektangulärt schema av rutor. Vissa rutor innehåller dolda minor. Det totala antalet minor är förutbestämt. Minorna är utlottade med obundet slumpmässigt urval, OSU, av rutor. Varje ruta har därför lika stor sannolikhet att bli minerad. Spelet går ut på att identifiera de rutor där minor gömmer sig. Om en minerad ruta öppnas är spelet över och förlorat. Om en tom ruta öppnas kommer en siffra att visas i rutan. Denna anger hur många minor det finns bland rutans grannar. Med granne menas angränsande ruta. De rutor som inte tillhör kanten på spelplanen har därför åtta grannar. När alla minors koordinater är identifierade och övriga rutor öppnade är spelet avklarat.

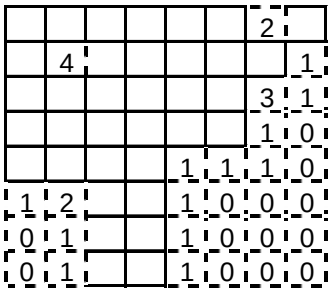
I de flesta situationer kan det logiskt avgöras om en ruta på spelplanen är tom eller minerad. Dock inte alltid. Vi skulle då vilja veta vilken av rutorna som har minst sannolikhet att innehålla en mina. Med en generell metod för att beräkna dessa sannolikheter kan vi i varje situation minimera risken att vid nästa klick stöta på en mina. Att utarbeta en sådan metod är en central del i vad mitt examensarbete behandlar. Det är dessutom grunden för problemställningarna: *Kan metoden användas för att simulera partier av MS röj, hur ofta kommer spelet att ”gå ut” och spelar det någon roll för utgången vilken ruta man börjar med?*

Arbetet inleds i avsnitt 2 med några exempel på hur man för hand kan beräkna sannolikheter att rutor innehåller minor. I avsnitt 3 förklaras den generella metod som använts för att kunna beräkna dessa sannolikheter. Ekvationssystem för binära variabler kommer då att spela en viktig roll. I avsnitt 4 förklaras hur metoden används för att kunna simulera spel i MS röj. Slutsatser och resultat av de simuleringar som gjorts redovisas i avsnitt 5. Slutligen, i avsnitt 6, förs en diskussion kring arbetet i allmänhet.

2 Några exempel

Huvudproblemet utgörs av att finna en generell metod att beräkna sannolikheten huruvida en ruta innehåller en mina. En sådan sannolikhet kommer i fortsättningen att benämnas *minsannolikhet*. För att erhålla en ökad förståelse för problemet behandlas i detta avsnitt några exempel.

Nedan ges en möjlig situation i MS röj:



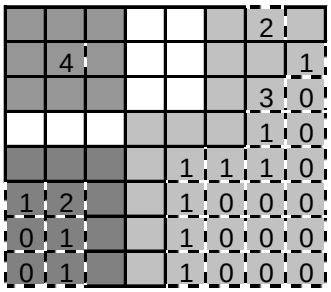
figur 1. En tänkbar situation i MS röj

Siffrorna i de öppnade rutorna säger endast något om dess grannar och ger ingen information angående övriga rutor. För de rutor som inte är granne till någon av de öppnade rutorna finns ingen specifik information och de har därför samma minsannolikhet.

För att göra det tydligt har nollor skrivits ut, vilket betyder att ingen mina finns bland rutans grannar. I de flesta versioner av MS röj står rutorna istället tomma.

För att underlätta beräkningarna kan varje situation på spelplanen delas upp i så kallade öar. En *ö* är en samling rutor så att om man vet antalet minor på den, kan alla minsannolikheter för öns rutor bestämmas via informationen inom öns gränser. En *oberoende ö* är en samling rutor så att informationen inom öns gränser är tillräcklig för att bestämma alla minsannolikheter på ön. Notera att rutorna på en ö ej behöver vara angränsande, vilket namnet kan tyckas antyda.

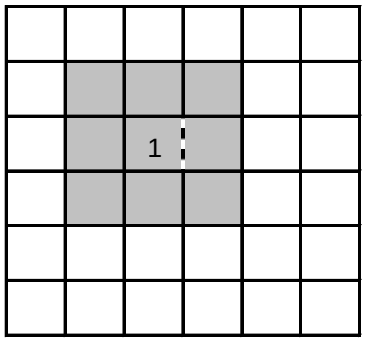
Nedan visas ett exempel på hur figur 1 kan uppdelas i öar:



Figur 2. Uppdelning i öar

Om antalet minor inom en ö är bestämt, kommer minsannolikheterna för öns rutor att vara helt bestämda av detta antal. Ofta finns det flera möjligheter för hur många minor en ö kan innehålla. Då kommer minsannolikheterna att vara mer svårberäknande. Exempel på båda fallen kommer att ges.

Exempel 1. Antag att situationen ser ut som följande:



Inom det skuggade området finns enbart en mina. Därför kan minsannolikheterna för öns rutor beräknas för sig, det vill säga öns minsannolikheter är oberoende av hur situationen i spelet är för övrigt. Alla rutor har dessutom samma förutsättningar varför de har lika stor minsannolikhet, 1/8. Jag kommer på nästa sida även att beräkna denna minsannolikhet på ett annat, mer generellt sätt, som nyttjas vid mer avancerade problem.

Antag att fem rutor är minerade. Eftersom det endast finns en mina utplacerad på ön måste det finnas fyra stycken bland övriga 27 rutor. Fyra minor kan vara utplacerade på C(27,4) olika sätt, där

$$C(a,b) = \binom{a}{b} = \frac{a!}{(a-b)!b!} = \frac{a(a-1)...(a-b+1)}{b!}$$

och beskriver antalet kombinationer av b element bland a, se Blom (1984a, kap 2.7).

Varje kombination har lika stor sannolikhet att inträffa varför vi kan säga att vi har en likformig sannolikhetsfördelning.

Sannolikheten räknas nu ut med hjälp av ”*den klassiska sannolikhetsdefinitionen*” som säger att vid likformig sannolikhetsfördelning är sannolikheten för en händelse lika med kvoten mellan antalet för händelsen gynnsamma utfall och antalet möjliga utfall, se Blom (1984a, s 28).

Låt A vara händelsen att en godtycklig men bestämd ruta *utanför* ön innehåller en mina. För att A ska inträffa måste de tre övriga minorna vara utplacerade bland de 26 kvarvarande rutorna. Detta kan ske på C(26,3) sätt, vilket också är antalet gynnsamma utfall för händelse A. Antalet möjliga utfall är enligt det tidigare C(27,4). Sannolikheten för att den godtyckliga rutan innehåller en mina blir därför

$$\mathrm{P(A)} = \frac{C(26,3)}{C(27,4)} = \frac{2600}{17550} = \frac{4}{27} \approx 0.148$$

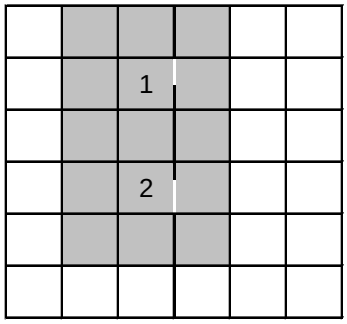
Låt B vara händelsen att en godtyckligt men bestämd ruta *inom* öns gränser innehåller en mina. Om B inträffar innebär det att resterande rutor på ön är tomma. Antalet gynnsamma utfall för händelsen B är därför på hur många sätt noll minor kan placeras ut bland sju rutor, det vill säga C(7,0). Antalet möjliga utfall är på hur många sätt en mina kan placeras ut på ön, det vill säga C(8,1). Sannolikheten för att den godtyckliga rutan ska innehålla en mina blir därför

$$\mathrm{P(B)} = \frac{C(7,0)}{C(8,1)} = \frac{1}{8} = 0.125$$

För att minimera risken att direkt stöta på en mina bör därför en ruta innanför öns gränser klickas på därnäst.

□

Exempel 2. Antag att situationen är som i figuren och att tio rutor är minerade.



Rutorna inom det skuggade området kan totalt innehålla två eller tre minor. Vi måste beräkna på hur många sätt tio minor kan placeras ut så att två respektive tre rutor på ön är minerade.

Antag att det finns två minor på ön. Då måste ettan och tvåan ”dela på” en mina i mittraden. Denna mina kan placeras på C(3,1) sätt. Den andra minan finns då i någon av de andra rutor som angränsar till tvåan. Den kan placeras ut på C(5,1) olika sätt. Nu finns åtta minor kvar att placera utanför ön. Detta kan ske på C(21,8) sätt. Multiplicerar vi dessa möjligheter får vi antal sätt att placera ut tio minor med två minor inom öns gränser.

Enligt samma princip kan man räkna ut antal sätt att placera ut tio minor med tre minerade rutor på ön. Det finns C(5,1)C(5,2)C(21,7) olika sätt. Alla dessa utfall har lika stor sannolikhet varför sannolikheten för två respektive tre minor kan beräknas enligt den klassiska sannolikhetsdefinitionen.

$$P(2\text{ minor}) = \frac{C(3,1)C(5,1)C(21,8)}{C(5,1)C(5,2)C(21,7) + C(3,1)C(5,1)C(21,8)} = \frac{3052350}{8866350} = \frac{21}{61} \approx 0.344$$

$$P(3\,\text{minor}) = \frac{C(5,1)C(5,2)C(21,8)}{C(5,1)C(5,2)C(21,7) + C(3,1)C(5,1)C(21,8)} = \frac{5814000}{8866350} = \frac{40}{61} \approx 0.656$$

Vi ser att summan av de två sannolikheterna blir 1.

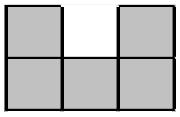
Minsannolikheten för en specifik ruta kan beräknas enligt samma resonemang. Vi undersöker på hur många sätt minorna kan vara utplacerade med bivillkoret att det är en mina i den specifika rutan, det vill säga beräknar antalet gynnsamma utfall, och dividerar med det totala antalet sätt att utplacera minorna på.

Låt A vara händelsen att en godtycklig men bestämd ruta bland öns mittrutor, se figur 3, innehåller en mina. Om A inträffar betyder det att ettan och tvåan ”delar på” en mina varför det endast kan vara två minor inom öns gränser. Händelsen A kan då inträffa på C(5,1)C(21,8) sätt varför sannolikheten att en mina finns i den godtyckliga rutan blir :

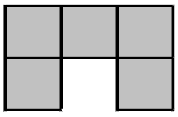
$$P(A) = \frac{C(5,1)C(21,8)}{C(5,1)C(5,2)C(21,7) + C(3,1)C(5,1)C(21,8)} = \frac{1017450}{8866350} = \frac{7}{61} \approx 0.115$$

Låt B och C vara händelserna att en godtycklig men bestämd ruta i figur 4 respektive figur 5 är minerad. Nedan följer dessa beräkningar:

figur 3.
Öns mittrutor.



figur 4.
Öns nedersta rutor.



figur 5.
Öns översta rutor.

$$P(B) = \frac{C(4,1)C(5,1)C(21,7) + C(3,1)C(21,8)}{C(5,1)C(5,2)C(21,7) + C(3,1)C(5,1)C(21,8)} = \frac{2936070}{8866350} = \frac{101}{305} \approx 0.331$$

$$P(C) = \frac{C(5,2)C(21,7)}{C(5,1)C(5,2)C(21,7) + C(3,1)C(5,1)C(21,8)} = \frac{1162800}{8866350} = \frac{8}{61} \approx 0.131$$

Låt D vara händelsen att en godtycklig men bestämd ruta utanför ön innehåller en mina. Eftersom det finns tio minor totalt, är summan av alla minsannolikheter 10, det vill säga

$$3P(A)+5P(B)+5P(C)+21P(D)=10$$

Det följer att

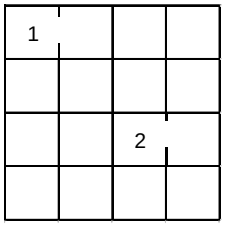
$$P(D)=\frac{10-3P(A)-5P(B)-5P(C)}{21}=\frac{64}{183}\approx 0.350$$

Vi bör i denna situation välja att klicka på någon av rutorna i figur 3.

□

Exempel 3. I de två tidigare exemplen har vi räknat nästan rent kombinatoriskt. I detta exempel skall vi räkna mer algebraiskt.

Antag att situationen ser ut som på bilden till vänster och att tre minor finns gömda bland spelplanens rutor. Den högra bilden visar rutornas numrering.



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Låt p_i vara minsannolikheten för ruta i, i=1,2…,16, och låt

$$t_i=\begin{cases}1, \text{om ruta } i \text{ är minerad}\\ 0, \text{för övrigt}\end{cases} \qquad i=1,2,...,16$$

Det gäller att $E(t_i)=p_i$ och härav $E(\sum_{i=1}^{16} t_i)=\sum_{i=1}^{16} p_i$.

Eftersom det totalt finns tre minor på spelplanen, det vill säga $\sum_{i=1}^{16} t_i = 3$, erhålls att $\sum_{i=1}^{16} p_i = 3$

Situationen ger då följande ekvationer:

$$\begin{cases} \sum_{i=1}^{16} p_i = 3 \\ p_1 = p_{11} = 0 \\ p_2 + p_5 + p_6 = 1 \\ p_6 + p_7 + p_8 + p_{10} + p_{12} + p_{14} + p_{15} + p_{16} = 2 \\ p_3 = p_4 = p_9 = p_{13} \\ p_2 = p_5 \\ p_7 = p_8 = p_{10} = p_{12} = p_{14} = p_{15} = p_{16} \end{cases}$$

Vi har 15 ekvationer men 16 obekanta storheter. Se p_6 som en fri variabel. Om vi vet den kan alla andra obekanta storheter bestämmas. Tyvärr är p_6 okänd. För att bestämma den måste vi beräkna antal kombinationer där ruta sex är minerad, $N_6(1)$, samt antal kombinationer där den ej är minerad, $N_6(0)$. Då är

$$p_6 = P(t_6 = 1) = \frac{N_6(1)}{N_6(0) + N_6(1)}$$

Vi kan via liknande resonemang som i föregående exempel beräkna $N_6(1)$ och $N_6(0)$:

$$N_6(1) = C(4,1)C(7,1) = 28 \text{ och } N_6(0) = C(2,1)C(7,2) = 42$$

Det följer att $p_6 = 28/70 = 2/5$. Vi kan sedan beräkna samtliga minsannolikheter. De blir:

$$p_2 = p_5 = \frac{3}{10}$$
$$p_3 = p_4 = p_9 = p_{13} = \frac{1}{10}$$
$$p_7 = p_8 = p_{10} = p_{12} = p_{14} = p_{15} = p_{16} = \frac{8}{35}$$

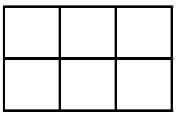
□

Teorin i exempel 3 för oss in på den matematiska metod jag utvecklat och använt mig av vid programmeringen. Istället för att använda ekvationer med p_i, kan de binära variablerna t_i användas som obekanta. Det totala antalet lösningar kommer då att motsvara antalet sätt minorna kan vara utplacerade på. Detta behandlas i avsnitt 3.

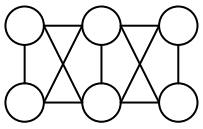
3 Matematisk teori

För stora spelplaner med invecklade situationer kan det vara mödosamt att beräkna minsannolikheter enligt metoderna i avsnitt 2. Vi kommer i detta kapitel att redovisa en matematisk metod som generellt kan hantera alla situationer på spelplanen.

Varje situation i MS röj kan åskådliggöras med hjälp av en graf bestående av noder och kanter. Noder representerar rutor och kan därför vara tomma eller minerade. Kanter beskriver vilka rutor som är grannar med varandra. Nedan följer ett exempel på en liten spelplan i MS röj och dess graffframställning.



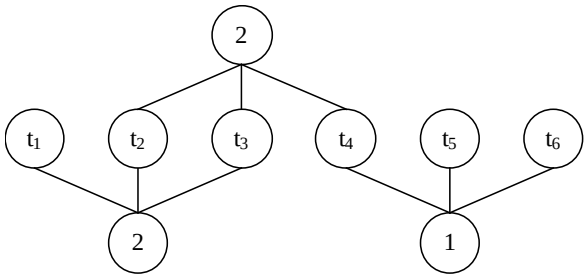
Figur 6. Spelplan i MS röj.



Figur 7. Graffframställning av figur 6.

För att på bästa sätt kunna beskriva den matematiska metoden kommer första exemplet att utgå från en graf. I grafen kan nämligen vissa kanter tas bort och problemet blir enklare och lättare att förstå. I exempel 2 återgår vi till rutschemat och visar ett fall där metoden är särskilt effektiv. Ett mer invecklat fall beskrivs i exempel 3 där rutorna delas upp i oberoende öar för att effektivisera programmeringen.

Exempel 1. Antag följande graf:



Antag att det finns tre minor totalt och låt

$$t_i = \begin{cases} 1, \text{om nod } i \text{ är minerad} \\ 0, \text{för övrigt} \end{cases} \qquad i = 1,2,...,6$$

Siffrorna i noderna säger hur många minor det finns bland nodens grannar och får inte förväxlas med variabeln t_i. Lägg märke till att grafens utseende *ej* kan beskrivas som en verklig situation i MS röj. Den är förenklad genom att använda färre kanter. Lösningens princip är dock densamma.

Grafens information samt vetskapen om det totala antalet minor kan skrivas som ett ekvationssystem,

$$\begin{cases} t_1 + t_2 + t_3 = 2 \\ t_2 + t_3 + t_4 = 2 \\ t_4 + t_5 + t_6 = 1 \\ t_1 + t_2 + t_3 + t_4 + t_5 + t_6 = 3 \end{cases}$$

vilket på matrisform kan skrivas som **At=b**, där

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 2 \\ 2 \\ 1 \\ 3 \end{bmatrix}$$

Detta ekvationssystem innehåller all information om grafen. Varje binär lösning är exempel på hur minorna kan vara utplacerade. Vi ska försöka hitta alla binära lösningar för att få fram antalet möjliga fall.

Den allmänna lösningen kan skrivas som $t = t_0 + c_1 z_1 + ... + c_6 z_6$, där t_0 är en partikulärlösning till **At=b**, $z_1,...,z_6$ en bas i nollrummet till **A** och $c_1,...,c_6$ skalärer. Den allmänna lösningen innehåller i vårt fall oändligt många lösningar. Det kan därför bli svårt att sålla bland dessa för att hitta de binära lösningarna. Vi måste gå tillväga på ett annat sätt.

Låt **C** vara den matris som fås om **b** läggs till som sista kolumn i **A**, det vill säga **C** = [**A**,**b**]. Matrisen **C** kan då tolkas som vårt ekvationssystem **At**=**b**.

Gauss Jordan elimination på **C**, se Anton (1994, p 12), ger den *trappstegsreducerade* matrisen $\mathbf{C^1}$. Detta är ett betydande steg i strävan att ta fram de binära lösningarna så effektivt som möjligt. Eliminationen reducerar matrisen till en enklare form men ändrar inte lösningsrummet.

$$\mathbf{C^1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\mathbf{C^1}$ representerar följande ekvationer:

$$\begin{cases} t_1 + t_5 + t_6 = 1 \\ t_2 + t_3 - t_5 - t_6 = 1 \\ t_4 + t_5 + t_6 = 1 \end{cases}$$

Eftersom t_1, t_2 och t_4 motsvarar de ledande ettorna i den radreducerade matrisen $\mathbf{C^1}$ kallas dessa för *ledande* variabler. Resterande variabler t_3, t_5 och t_6 kallas för *fria* variabler. Löser vi de ledande variablerna i termer av de fria variablerna erhålles

$$\begin{cases} t_1 = 1 - t_5 - t_6 \\ t_2 = 1 - t_3 + t_5 + t_6 \\ t_4 = 1 - t_5 - t_6 \end{cases} \qquad (3.1)$$

Om vi kan testa alla binära kombinationer av de fria variablerna, ger det värden på de ledande variablerna. Behåller vi de lösningar där alla värden är binära, det vill säga även där de ledande variablerna är binära, får vi alla möjliga lösningar. Skriv om (3.1) på matrisform enligt $\mathbf{t^1=De}$, där

$$\mathbf{t^1} = \begin{bmatrix} t_1 \\ t_2 \\ t_4 \end{bmatrix} \qquad \mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \qquad \mathbf{e} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ t_3 \\ 0 \\ t_5 \\ t_6 \end{bmatrix}.$$

Konstanterna från (3.1) ses i första kolumnen i **D**. Om binära värden sätts in på de fria variablerna i **e** kommer produkten **De** att ge värden på de ledande variablerna t_1, t_2 och t_4. Värdena på variablerna i $\mathbf{t^1}$ och **e** bildar då tillsammans en lösning till **At=b**. Denna kan, men behöver inte vara binär. För att hitta alla binära lösningar skapas, utifrån vektorn **e**, en matris **E** med 2^3 kolumner. Varje kolumn representerar en unik binär kombination av de fria variablerna t_3, t_5 och t_6. Övriga variabler sätts till noll.

$$\mathbf{E} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

I matrisprodukten **T**=**DE** står varje kolumn *j* för en kombination av de ledande variablerna t_1, t_2, och t_4 som ges av den binära uppsättningen på de fria variablerna från kolumn *j* i **E.**

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 2 & 1 & 2 & 1 & 3 & 2 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$

Sammanfoga **T** med de binära kombinationer på de fria variablerna som motsvarar respektive lösning i **T** . Kalla den sammanfogade matrisen **T2.** Varje kolumn i **T2** är en lösning på våra obekanta variabler $t_1,\ldots,t_6$.

$$\mathbf{T2} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 2 & 1 & 2 & 1 & 3 & 2 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Behåll enbart de binära lösningarna, det vill säga kolumnerna 1, 2, 4 och 6. Detta kan göras genom att multiplicera varje element i **T2** med sig självt och behålla de kolumner där inga värden ändras. Ty talen 0 och 1 skiljer sig åt från alla andra tal på så sätt att de förblir oförändrade efter att de har multiplicerats med sig själva. Vi erhåller då en ny matris där varje kolumn är en binär lösning $\mathbf{\Pi}_i$, $i = 1,\ldots,4$, till ekvationssystemet **At=b.** Transponera matrisen för att få lösningarna $\mathbf{\Pi}_1,\ldots,\mathbf{\Pi}_4$ radvis.

	t_1	t_2	t_3	t_4	t_5	t_6
Π_1	1	1	0	1	0	0
Π_2	1	0	1	1	0	0
Π_3	0	1	1	0	1	0
Π_4	0	1	1	0	0	1

Sannolikheten att ruta *i* innehåller en mina är lika med andelen lösningar $\mathbf{\Pi}_1,\ldots,\mathbf{\Pi}_4$ där t_i=1. Vi kan beräkna dessa sannolikheter genom att summera alla kolumner i tabellen och dividera med antal rader. Låt p_i, i=1,…,6, vara sannolikheten att ruta i är minerad, det vill säga p_i = P(t_i=1). Då får vi en minsannolikhetsvektor enligt :

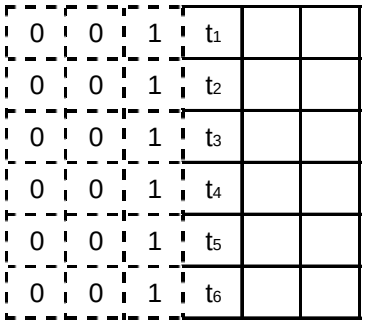
p_1	p_2	p_3	p_4	p_5	p_6
0.5	0.75	0.75	0.5	0.25	0.25

För att minimera risken att direkt stöta på en mina bör således någon av rutorna nummer 5 eller 6 väljas därnäst.

□

Vi har nu i exempel 1 tagit upp ett problem som även kan lösas med enkel kombinatorik. Den matematiska metod som används kan tyckas onödig. Men kom ihåg att metoden är generell och kan lösa varje situation på spelplanen. Det finns situationer för vilka den är särskilt effektiv. Nedan kommer ett sådant exempel.

Exempel 2. Vi Återvänder till MS röjschemat. Antag att situationen är enligt följande:



Vi vill veta minsannolikheterna för rutorna med variablerna t_i, i=1,2,…,6. Rutorna till vänster om kolumnen med ettor är öppnade. Situationen ger ett ekvationssystem i form av matrisen:

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Gauss-Jordan elimination ger matrisen:

$$\mathbf{C^1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Vi ser att alla sex obekanta variabler t_i, i=1,…,6 är ledande. Det betyder att vi får en entydig lösning på dessa. Sannolikhetsvektorn är då identisk med denna lösning och blir:

p_1	p_2	p_3	p_4	p_5	p_6
0	1	0	0	1	0

I den aktuella situation behöver således ingen chansning göras, vilket man till en början skulle kunna tro.

□

Eftersom den binära lösningen i exempel 2 är entydig, är min metod där väldigt effektiv. I vissa situationer på spelplanen är dock metoden inte lika fördelaktig. Problem kan särskilt uppstå då de fria variablerna blir alltför många. 30 fria variabler ger till exempel 2^{30}, det vill säga över en miljard olika kombinationer att undersöka. Vi måste då hitta ett sätt att reducera onödiga lösningar. Ett sätt, att beräkna oberoende öar var för sig, beskrivs i exempel 3.

Exempel 3. Antag att situationen på spelplanen ser ut enligt:

1	t_2	t_3	t_4
t_5	t_6	t_7	1
t_9	t_{10}	t_{11}	t_{12}
t_{13}	t_{14}	t_{15}	t_{16}

Vi får ekvationerna:

$$\begin{cases} t_2 + t_5 + t_6 = 1 \\ t_3 + t_4 + t_7 + t_{11} + t_{12} = 1 \\ t_2 + t_3 + t_4 + t_5 + t_6 + t_7 + t_9 + t_{10} + t_{11} + t_{12} + t_{13} + t_{14} + t_{15} + t_{16} = 3 \end{cases}$$

Dessa anges av matrisen:

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 3 \end{bmatrix}$$

Gauss-Jordan elimination ger:

$$\mathbf{C^1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Vi ser att t_2, t_3 och t_9 är ledande variabler. Resterande kommer att vara fria. Löser vi de ledande variablerna i termer av de fria variablerna ger det:

$$\begin{cases} t_2 = 1 - t_5 - t_6 \\ t_3 = 1 - t_4 - t_7 - t_{11} - t_{12} \\ t_9 = 1 - t_{10} - t_{13} - t_{14} - t_{15} - t_{16} \end{cases}$$

Schemat kan presenteras på matrisform enligt $\mathbf{t^1}=\mathbf{De}$, där

$$\mathbf{t^1} = \begin{bmatrix} t_2 \\ t_3 \\ t_9 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & -1 & -1 \end{bmatrix}$$

$$\mathbf{e}^{\mathrm{T}} = \begin{bmatrix} 1 & t_1 & 0 & 0 & t_4 & t_5 & t_6 & t_7 & t_8 & 0 & t_{10} & t_{11} & t_{12} & t_{13} & t_{14} & t_{15} & t_{16} \end{bmatrix}$$

För att reducera antalet binära lösningar kan spelplanen delas upp i tre öar, se figur nedan. Dessa är oberoende av varandra och även av resterande rutor eftersom antalet minor inom varje ö är bestämt. Vi kan beräkna minsannolikheterna inom varje oberoende ö var för sig.

1	t_2	t_3	t_4
t_5	t_6	t_7	1
t_9	t_{10}	t_{11}	t_{12}
t_{13}	t_{14}	t_{15}	t_{16}

Men hur skall datorn kunna skilja öarna åt? Låt oss se på den radreducerade matrisen

$$\mathbf{C^1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

En grupp av rader som endast innehåller ettor som ingen annan rad innehåller är en oberoende ö. Vi ser här att alla tre rader är oberoende öar. Vi kan dela upp ekvationssystemet **T**=**DE** i tre separata delar och lösa varje del var för sig enligt tidigare princip. Matrisekvationen t_2=$\mathbf{D^1e^1}$ står då för den första öns relationer mellan variablerna, där

$$\mathbf{D^1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$(\mathbf{e^1})^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & t_5 & t_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\mathbf{D^1}$ är första raden i **D** och $\mathbf{e^1}$ är **e** utan de variabler som är beroende av andra öar.

Om värden sätts in på de fria variablerna i $\mathbf{e^1}$, kommer multiplikationen $\mathbf{D^1e^1}$ att ge ett värde på den ledande variabeln t_2. Det värdet kombinerat med de fria variablerna som gav t_2 blir då en lösning på våra obekanta variabler. För att få alla lösningar skapas en matris $\mathbf{E^1}$ med 2^2 kolumner. Varje kolumn representerar en unik binär kombination av de fria variablerna t_5 och t_6. $\mathbf{E^1}$:s dimensioner kommer att vara $17x2^2$. Vi behöver således testa fyra kombinationer av de fria variablerna för att få fram de binära lösningarna för den första öns ekvationer. Vi kan då bestämma minsannolikheterna för rutorna i den ön.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}
0	$\frac{1}{3}$	0	0	$\frac{1}{3}$	$\frac{1}{3}$	0	0	0	0	0	0	0	0	0	0

På samma sätt kan de två andra öarnas minsannolikheter beräknas. De öarna består av fyra respektive fem fria variabler. Vi behöver därför testa 2^4 respektive 2^5 kombinationer av de fria variablerna för att få fram de binära lösningarna.

Eftersom matrisernas dimension behålls trots uppdelningen, det vill säga de variabler utanför respektive ö tas inte bort utan sätts till noll, kommer övriga rutors minsannolikheter att vara noll. På så sätt blir minsannolikhetsvektorn lika stor för alla öar. Vi kan addera dessa vektorer för att få fram samtliga minsannolikheter. De andra öarnas sannolikhetsvektorer blir enligt följande:

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}
0	0	$\frac{1}{5}$	$\frac{1}{5}$	0	0	$\frac{1}{5}$	0	0	0	$\frac{1}{5}$	$\frac{1}{5}$	0	0	0	0

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}
0	0	0	0	0	0	0	0	$\frac{1}{6}$	$\frac{1}{6}$	0	0	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

Addering av de olika öarnas sannolikhetsvektorer ger lösningen.

□

Totalt måste vi testa $2^2 + 2^4 + 2^5 = 52$ kombinationer för att få minsannolikheterna. Om vi ej tar hänsyn till öarna behöver 2^{11} kombinationer testas. Antalet icke-binära lösningar har därmed reduceras avsevärt. Med andra ord, att dela upp spelplanen i oberoende öar effektiviserar.

4 Från teori till simulering

Detta avsnitt handlar om hur jag kan utnyttja den matematiska metod jag tidigare beskrivit för att simulera ett parti MS röj. Algoritmen återfinns i Appendix A. Källkoder hittas i Appendix B.

Låt oss anta att spelplanen är av storlek 4x4 och att tre rutor är minerade. Rutorna är numrerade enligt följande:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

4.1 Utökad grannmatris

I grannmatrisen beskrivs vilka rutor som är grannar med varandra. Den *utökade grannmatrisen*, se bild nedan, är grunden för de ekvationssystem som behövs för att beräkna minsannolikheter under spelets gång.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	–
2	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	–
3	0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	–
4	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	–
5	1	1	0	0	0	1	0	0	1	1	0	0	0	0	0	0	–
6	1	1	1	0	1	0	1	0	1	1	1	0	0	0	0	0	–
7	0	1	1	1	0	1	0	1	0	1	1	1	0	0	0	0	–
8	0	0	1	1	0	0	1	0	0	0	1	1	0	0	0	0	–
9	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0	0	–
10	0	0	0	0	1	1	1	0	1	0	1	0	1	1	1	0	–
11	0	0	0	0	0	1	1	1	0	1	0	1	0	1	1	1	–
12	0	0	0	0	0	0	1	1	0	0	1	0	0	0	1	1	–
13	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	–
14	0	0	0	0	0	0	0	0	1	1	1	0	1	0	1	0	–
15	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	1	–
16	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	–
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3

Rad *i*, *i*=1,…,16, visar ettor i den kolumn *j*, *j*=1,…,16, där ruta *i* har *j* som grannruta, nollor för övrigt. Den yttersta kolumnen är till en början tom men kommer så småningom att fyllas

med information i form av siffror från de öppnade rutorna. Nedersta raden beskriver ekvationen att det totalt finns tre minor, det vill säga $\sum_{i=1}^{16} t_i = 3$.

4.2 Minutlottning

Minorna slumpas ut med hjälp av Matlabs slumpgenerator, se Melin (1996, p 58). Slumpgeneratorn har en cykel på 2^{1492} . För att starta processen behövs ett frö i form av en siffra. Om användaren ej ger ett frö så startar Matlab med värdet noll.

Utlottningen sker med OSU om tre rutor bland 16. Varje ruta har då lika stor sannolikhet att innehålla en mina. En tänkbar minutlottning skulle kunna se ut enligt:

1			1
	1		

Vi ser att ruta nummer 1, 4 och 6 är minerad. Utifrån detta görs en *grannminmatris*.

1	2	2	0
2	1	2	1
1	1	1	0
0	0	0	0

Grannminmatrisen visar för varje ruta hur många minor det finns bland dess grannrutor. Den kommer att användas när rutorna öppnas.

4.3 Hur spelet börjar

Inför första valet har alla rutor lika stor minsannolikhet, varför vi slumpar vilken ruta som ska väljas. Om den innehåller en mina är spelet förlorat. Är rutan ej minerad får spelaren veta hur många minor det finns bland rutans grannar. Den informationen finns i grannminmatrisen och

sätts in på den rad som representerar den öppnade rutan i den utökade grannmatrisens informationskolumn. Antag att ruta 8 väljs. Den har *en* minerad grannruta, vilket gör att vi infogar en etta på rad 8 i informationskolumnen. Ruta 8 är inte längre misstänkt för att vara minerad, varför den rutan inte ska vara med i de fortsatta sannolikhetsberäkningarna. Alla värden i kolumn 8 i den utökade grannmatrisen sätts därför till noll. Bilda en matris av de rader som innehåller information i informationskolumnen, det vill säga rad 8 samt sista raden.

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 3 \end{bmatrix}$$

Matrisen står för följande ekvationer:

$$\begin{cases} t_3 + t_4 + t_7 + t_{11} + t_{12} = 1 \\ t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7 + t_9 + t_{10} + t_{11} + t_{12} + t_{13} + t_{14} + t_{15} + t_{16} = 3 \end{cases}$$

Ekvationssystemet ovan innehåller all information som den aktuella situationen på spelplanen ger.

4.4 Spelet fortsätter

Minsannolikheterna beräknas för den aktuella situationen med hjälp av den teori som beskrivits i avsnitt 3. Rutan med minst minsannolikhet väljs. I de fall där flera rutor har samma minsta minsannolikhet sker lottning bland dessa. Antag att ruta *j* väljs. Om den rutan är minerad, är spelet förlorat. Om så inte är fallet, får spelaren veta hur många minor det finns bland rutans grannar. Den informationen finns i grannminmatrisen och sätts in på rad *j* i informationskolumnen i den utökade grannmatrisen. Alla värden i den utökade grannmatrisen i kolumnen *j* sätts till noll. De rader som innehåller information bildar en ny matris och nya minsannolikheter kan beräknas. Proceduren fortsätter till dess att alla minor är lokaliserade, det vill säga när tre av rutorna har minsannolikhet ett, resterande har minsannolikhet noll.

4.5 En simuleringsstudie

Låt oss tillsammans följa datorn i ett simulerat spel MS röj. Spelplanen är av storlek 4x4 med totalt tre minor.

Minorna slumpas ut enligt följande:

			*
	*		
	*		

Minornas placering är obekant för spelaren. Första klicket lottas bland spelplanens alla rutor. Ruta nummer 8 väljs. Den är inte minerad och har *en* granne som innehåller en mina. Spelplanens utseende ges nedan. Till höger visas minsannolikheter för respektive ruta.

			1

0.2	0.2	0.2	0.2
0.2	0.2	0.2	0
0.2	0.2	0.2	0.2
0.2	0.2	0.2	0.2

Alla rutor som är oöppnade har här lika stor sannolikhet att innehålla en mina. Detta inses med hjälp av enkel kombinatorik. Ruta nummer 10 väljs efter lottning. Detta ger följande:

			1
	2		

0.148	0.148	0.185	0.185
0.259	0.259	0.222	0
0.259	0	0.222	0.185
0.259	0.259	0.259	0.148

Situationen på spelplanen är nu mer invecklad eftersom flera av rutorna beror både av ettan och av tvåan. Enligt de beräkningar som görs är ruta nummer 1, 2 och 16 de med lägst minsannolikhet. Ruta nummer 2 väljs efter lottning.

	1		
			1
	2		

0.125	0	0.187	0.125
0.281	0.281	0.125	0
0.281	0	0.187	0.125
0.281	0.281	0.281	0.187

Ruta nummer 7 väljs efter lottning.

	1		
		2	1
	2		

0	0	0	0.444
0	1	0	0
0.222	0	0.111	0.444
0.222	0.222	0.222	0.111

Vi ser till höger att ruta nummer 6 är minerad. Det är då två minor kvar att identifiera. Ruta 5 väljs efter lottning. Denna har minsannolikhet noll och innehåller därför ingen mina.

	1		
1	*	2	1
	2		

0	0	0	0.428
0	1	0	0
0	0	0.143	0.428
0.286	0.286	0.286	0.143

Ruta nummer 9 väljs efter lottning.

	1		
1	*	2	1
2	2		

0	0	0	0.5
0	1	0	0
0	0	0	0.5
0.5	0.5	0	0

Vi kan nu se att vi börjar närma oss lösningen. De två återstående minorna är nu lokaliserade till någon av de fyra rutor som har minsannolikhet 0.5. Lägg märke till att det fortfarande finns oöppnade rutor där vi vet att de ej innehåller minor. Lottning ger att ruta nummer 3 väljs.

	1	2	
1	*	2	1
2	2		

0	0	0	1
0	1	0	0
0	0	0	0
0.5	0.5	0	0

Två av minorna är nu identifierade. Den sista minan finns i ruta nummer 13 eller 14. Vi kan dock istället klicka på någon av de rutor som har minsannolikhet noll. Ruta nummer 11 väljs efter lottning.

	1	2	*
1	*	2	1
2	2	2	

0	0	0	1
0	1	0	0
0	0	0	0
0	1	0	0

Vi ser i det högra rutsystemet att samtliga minsannolikheter är binära tal. Minornas placeringar är därmed fastställda och spelet avklarat.

□

Min vision var att programmera en variant av spelet MS röj, där användaren i varje situation själv väljer ruta med hjälp av successivt beräknade minsannolikheter enligt exemplet ovan.

Tyvärr fanns inte tid att göra verklighet av visionen varför jag fick nöja mig med att simulera spel. Ingen grafik visas i simuleringarna. Bilderna ovan är illustrerade i efterhand genom att gå in i programmet och översätta kod till bild.

5 Simuleringsresultat

I detta avsnitt kommer de resultat som erhölls vid simuleringarna att redovisas. Simuleringarna är gjorda i Matlab. På grund av minnesutrymme har bara spel med spelplansstorleken 4x4 simulerats. Intressant är att veta huruvida det första klickets position har någon betydelse för utgången av spelet. Därför gjordes fyra olika typer av simuleringar beroende på hur första klicket slumpades ut.

1. Var som helst på spelplanen(F).
2. Bland spelplanens åtta kantrutor(K).
3. Bland spelplanens fyra hörnrutor(H).
4. Bland spelplanens fyra mittrutor(M).

Jag kommer i fortsättningen att försöka hänvisa till de olika simuleringstyperna genom att använda bokstavsbeteckningarna: F, K, H och M. I vissa fall kommer dock sifferbeteckningarna att användas.

Tabell 1. Resultat 4000 simuleringar. Tre minor.

Simuleringstyp	F	K	H	M
y	0.67575	0.65675	0.70150	0.67500
m	2.9352	2.9315	2.9095	2.9346

y=andel lyckade spel, m=medelantal minor kvar att upptäcka om spelet misslyckas

Vi ser att vid de flesta gånger spelet misslyckas, återstår samtliga minor att röja. Det första klicket, vilket har en minsannolikhet på 3/16, är en stor del av förklaringen. Högst sannolikhet att klara spelet tycks simuleringstyp H ha. I avsnitt 5.2 kommer jag att presentera tester som bekräftar detta.

Tabell 2. Resultat 4000 simuleringar. Två minor.

Simuleringstyp	F	K	H	M
y	0.81775	0.82400	0.84750	0.80800
m	2	2	2	2

y=andel lyckade spel, m=medelantal minor kvar att upptäcka om spelet misslyckas

Även här tycks hörnrutorna vara bäst att börja med. Vi ser också att när vi väl har röjt den första minan tycks spelet alltid ”gå ut”.

Låt A vara händelsen att spelet ”går ut”. Låt X vara antal gånger som A inträffar i n simuleringar. Då kan X ses som en binomialfördelad stokastisk variabel, det vill säga

$$X \sim \mathrm{Bin}(n,p)$$

där *p* är sannolikheten att A inträffar i ett enskilt försök.

5.1 Intervallskattning

Konfidensintervall för *p* kan göras med utnyttjande av normalapproximation, se Blom (1984b, kap 20). Eftersom n är stort blir de approximativa konfidensintervallen bra.

Enligt centrala gränsvärdessatsen gäller för stora n approximativt att

$$\frac{X}{n} \in N(p, D) \quad \text{där } D = \sqrt{\frac{pq}{n}}$$

Ett approximativt konfidensintervall, I_p, för *p* blir då

$$I_p = (p^* - \lambda_{\alpha/2} d \, , \, p^* + \lambda_{\alpha/2} d)$$

där d är medelfelet, det vill säga $d = \sqrt{\frac{p^*(1-p^*)}{n}}$, λ_α är normalfördelningens kvantil, (1-α) är konfidensgraden och p^* är punktskattningen för *p*, dvs i vårt fall, andelen lyckade spel i de n simuleringarna.

Låt $X_i \in \text{Bin}(4000, p_i)$ vara stokastiska variabler från simuleringstyp *i* med tre minor, *i*=1,2,3,4. Här är p_i sannolikheten att ett simulerat spel ”går ut” .

Låt

$$p_1 = p_F$$
$$p_2 = p_K$$
$$p_3 = p_H$$
$$p_3 = p_M$$

Approximativt 95%-iga konfidensintervall för p_i blir då

Tabell 3. Konfidensintervall. Tre minor

I_{p_F}	0.67575±0.0146
I_{p_K}	0.65675±0.0148
I_{p_H}	0.70150±0.0142
I_{p_M}	0.67500±0.0146

Låt $Y_i \in \text{Bin}(4000, \pi_i)$ vara stokastiska variabler från simuleringstyp *i* med två minor, *i*=1,2,3,4. Här är π_i sannolikheten att ett simulerat spel ”går ut”.

Approximativt 95%-iga konfidensintervall för π_i blir då

Tabell 4. Konfidensintervall. Två minor

I_{π_F}	0.81775±0.0120
I_{π_K}	0.82400±0.0118
I_{π_H}	0.84750±0.0112
I_{π_M}	0.80800±0.0123

5.2 Hypotesprövning

För att kunna avgöra om $p_i \neq p_j$ och $s_i \neq s_j$, i ≠ j, kan parvisa hypotesprövningar utföras, se Blom (1984b, kap 21). Vi använder oss här liksom vid konfidensintervallskattningen av normalapproximation varför testerna är approximativa. Eftersom n är stort är vår approximation bra.

För två stokastiska variabler $X \in Bin(n_x,p_x)$ och $Y \in Bin(n_y,p_y)$, kan en hypotesprövning göras enligt följande:

Nollhypotesen H_0 och mothypotesen H_1 antas vara

$$H_0 : p_x = p_y \text{ och } H_1 : p_x \neq p_y$$

Antag att H_0 är sann och kalla det gemensamma värdet $p_x = p_y$ för p. En skattning av p ges då av

$$p^* = \frac{x+y}{n_x+n_y}$$

Vi kan då skapa en teststatistika u enligt

$$u = \frac{\left| p^*_x - p^*_y \right|}{d} \quad \text{där } d = \sqrt{p^* q^* \left(\frac{1}{n_x} + \frac{1}{n_y} \right)}$$

Om $u > \lambda_{\alpha/2}$ är skillnaden signifikant på nivån α. Låt $X_i \in Bin(4000,p_i)$ och $Y_i \in Bin(4000,\pi_i)$, enligt avsnitt 5.1.

Några av de tester som gjorts redovisas nedan.

Tabell 5, parvisa hypotesprövningar

Nollhypotes(H_0)	Teststatistika(u)
$p_K=p_M$	2.44
$p_K=p_H$	6.06
$p_F=p_H$	3.51
$\pi_K=\pi_H$	4.01
$\pi_F=\pi_H$	5.04
$\pi_H=\pi_M$	6.61
$\pi_K=\pi_M$	2.61

Låt $\alpha = 0.05$. för en nollhypotes där teststatistikan, u, är större än $\lambda_{0.025}=1.96$ kan den förkastas på 5%-nivån.

Tabellen visar att i alla tester som redovisas kan nollhypotesen förkastas. För båda typerna av spel kan fastställas att bästa öppningen görs i en hörnruta. För spel med tre minor är kantrutorna sämsta startalternativet. Sämsta öppningen för spel med två minerade rutor är i en mittruta. Jag ska nedan försöka mig på att förklara orsaken till detta.

En hörnruta har ganska stor sannolikhet att ej ha någon minerad granne. Om så är fallet kan dessa grannrutor öppnas utan risk för att vara minerade och på så sätt skaffa värdefull information angående övriga rutor. Detta kan vara en förklaring varför en start i hörnrutorna ger de bästa värdena.

Vid de flesta fall där spelet misslyckas, sker misstaget alldeles i början. Vi kan se det på variabeln m:s värden i tabell 1 och i tabell 2. Det är därför viktigt att få en bra start så att nästa klick är relativt säkert. Antag att spelplanen innehåller tre minor och att första klicket görs i en kantruta. Om den rutan visar en etta, det vill säga det finns en mina bland grannrutorna, kommer alla rutor ha samma minsannolikhet, 1/5, inför nästa klick. Ingen annan start kan ge ett sådant oönskat utfall. Detta kan förklara varför en start i kantrutorna har den lägsta sannolikheten att klara spelet.

När spelplanen innehåller två minor kan en öppning i mittrutorna medföra att vi måste välja en ruta med minsannolikhet 1/8. Detta sker om en etta visar sig i den rutan vi valt. Ingen annan öppning kan göra att vi i nästa klick måste välja en ruta med så stor minsannolikhet. En start i mittrutorna ger därför lägst sannolikhet att klara spelet.

6 Diskussion

I detta sista avsnitt behandlas diverse aspekter som till exempel brister i arbetet och eventuella kompletteringar som skulle kunna göras. Kopplingar till ett stort vetenskapligt område tas också upp.

- En stor del av mitt arbete utgjordes av att försöka lösa binära ekvationssystem så effektivt som möjligt. Med effektivt menar jag att vid ett tidigt skede kunna eliminera så många av de icke binära lösningarna som möjligt. I MS röj kan spelplanen vara mycket stor. I de flesta versioner begränsas dock denna till en storlek av 30x16. Att dela upp spelplanen i oberoende öar och fria variabler gör att ingen situation på spelplanen kräver fler än 30 obekanta variabler. Det innebär med andra ord 2^{30} olika kombinationer att testa. I Matlab, det vill säga det programmeringsspråk som använts tar minnet slut vid en sådan beräkning för cirka 20 variabler. Här uppstår ett problem. Kan variablerna delas upp ytterligare? Ett sätt är att omvandla de beroende öarna till oberoende genom att ansätta bivillkoret att x minor gömmer sig på ön. Genom att beräkna minsannolikheter för alla öar och alla dess möjliga bivillkor, kan de kombinationer som ger fel antal kvarvarande minor elimineras. Tyvärr har detta inte kunnat utföras och kan ses som ett potentiellt problem vid eventuella fortsatta studier i ämnet.

- Allt eftersom spelet pågår förändras utgångsläget för att beräkna minsannolikheterna. Öarna byter utseende och karaktär, vissa variabler kommer till medan andra försvinner. För att kunna följa denna utveckling krävs en flexibilitet i programmeringen som jag inte fullt behärskar. Spelplanen begränsades därför så kraftigt som till ett kvadratiskt rutsystem innehållande 16 rutor. Då kan alla variabler vara med till en början och ingen beräkning kräver mer än 2^{16} kombinationer att testa. Min ursprungliga vision var att kunna simulera spel av storleken 30x30.

- Är strategin att i alla situationer välja den ruta med minst minsannolikhet alltid den optimala? I vissa fall kan en situation uppstå där två rutor vardera har 0.5 i minsannolikhet. Alla rutor kring dessa är öppnade. Valet mellan dessa rutor måste förr eller senare ske. Det kan då vara bättre att så fort som möjligt gissa och på så sätt spara tid. Om målet är att komma så långt som möjligt bör dock dessa båda rutor sparas till sist. Ett annat fall som ifrågasätter min strategi är när en ruta med större minsannolikhet i snitt ger bättre information än en ruta med mindre sannolikhet. Kanske minsannolikheterna skiljer sig åt så pass lite att rutan med större sannolikhet är att föredra. I de fall där mitt program lottar vilken ruta som ska väljas skulle det naturligtvis vara bättre att analysera situationen och välja den ruta som mest troligt ger den bästa informationen. Strategin är alltså inte nödvändigtvis optimal, men den fungerar ändå relativt väl.

- I den tryckta vetenskapliga litteraturen har enbart en uppsats hittats som behandlar MS röj: Adamatzky (1997). Denna uppsats behandlar en metod att automatiskt lokalisera minor men ger inga resultat rörande minsannolikheter. Buzzard (1999) analyserar *en* situation i MS röj och beräknar minsannolikheter med hjälp av kombinatorik.

- Detta arbete kan tyckas vara fristående, men viss koppling finns till ett större och mer vetenskaplig område: ”Bayesian networks”, se till exempel Ripley (1996, Kap 8). Ett sådant nätverk är en graf med noder och kanter. Till varje nod hör en stokastisk variabel. Variablerna är beroende och beroendet preciseras delvis med hjälp av kanterna. Värdena för variablerna är bara kända för vissa noder. Det är då av intresse att till exempel kunna prediktera övriga noders variabelvärden. Denna prediktion sker normalt med hjälp av en speciell simuleringsmetod: Markov chain Monte Carlo, McMC, se ,till exempel, Ripley (1996, Appendix A3).

Referenser

Adamatzky, A. (1997). How cellular automaton plays minesweeper. Applied Mathematics and Computation 85, 127-137.

Anton, H. (1994). Elementary Linear Algebra 2nd ed. John Wiley & Sons, inc, New York

Blom, G. (1984a). Sannolikhetsteori med Tillämpningar. Studentlitteratur, Lund

Blom, G. (1984b). Statistikteori med Tillämpningar. Studentlitteratur, Lund

Buzzard (1999). Minesweeper: Advanced tactics, http://world.std.com/~buzzard/minesweeper/

Melin, B. (1996). Användarhandledning för MATLAB 4.2. Teknisk databehandling, Uppsala

Ripley, B.B. (1996). Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge

Appendix A Algoritm, spela ett parti Ms röj

1. Bestäm storlek på spelplanen och antal minor
2. Skapa en grannminmatris
3. Slumpa ut minorna
   - 3.1 Gör en grannminmatris
4. Slumpa första klicket
   - 4.1 Om minerad ruta
     - 4.1.1 Avsluta
   - 4.2 Om tom ruta
     - 4.2.1 Översätt informationen till grannminmatrisen
5. Så länge det finns minor kvar att identifiera
   - 5.1 Låt A vara matrisen bestående av de rader i grannminmatrisen som innehåller information
   - 5.2 Räkna ut minsannolikheterna för ekvationssystemet A
   - 5.3 Om någon ickevald ruta har minsannolikhet 1
     - 5.3.1 Spara rutans placering i en vektor
     - 5.3.2 Antal minor kvar = antal minor kvar –1
     - 5.3.3 Gå till 5.1
   - 5.4 Välj den ruta med minst minsannolikhet av de rutor jag ej tidigare valt
     - 5.4.1 Om minerad ruta
       - 5.4.1.1 Avsluta. Spara antal kvarvarande minor
     - 5.4.2 Om tom ruta
       - 5.4.2.1 Översätt informationen från rutan till grannmatrisen
       - 5.4.2.2 Spara rutans placering i en vektor
   - 5.5 Gå till 5
6. Spelet är klart

Appendix B Källkod

Tre program tas med i appendix. Det första, huvudprogrammet, simulerar ett antal Ms röj-spel och gör de beräkningar utifrån frågeställningarna som beskrivs i introduktionen. De två andra, delprogrammen ”minesweeper” och ”binary”, spelar ett parti Ms röj respektive löser ekvationssystem för binära variabler och beräknar minsannolikheter.

```
function x = simulering1(antsimuleringar,antminor,storlek);
%simulerar ett antal Ms röj spel och räknar
%ut hur ofta den klarar spelet. De gånger spelet inte "går ut" beräknas
%medeltalet minor det är kvar att röja

klara=0;                                    %nollställ antal klarade
                                            %spelomgångar
ejklarat=0;                                 %nollställ antal ej klarade
                                            %spelomgångar
msum=0;                                     %nollställ antal oupptäckta minor
for i=1:antsimuleringar                     %gör detta antsimuleringar gånger


   m = minesweeper4(antminor,storlek);      %spela ett parti Ms röj
   if m==0                                  %om spelet "gick ut"
       klara=klara + 1;                     %lägg till en till antal klarade
                                            %partier
   else                                     %om spelet ej "gick ut"
       ejklarat=ejklarat + 1;               %lägg till en till antal
                                            %ej klarade partier
       msum=msum+m;                         %lägg till hur många minor det
                                            %var kvar att röja
   end
end

klarade=(klara/antsimuleringar)*100         %räknar ut hur stor del av alla
                                            %spel som "gick ut"
ejklaratmedel=msum/ejklarat                 %av de gånger som spelet ej gick
                                            %ut så beräknas
                                            %i snitt hur många minor det då
                                            %finns kvar
```

```
function m = minesweeper(antminor,storlek);
%spela ett parti Ms röj


minefield=minefield2(storlek,storlek,antminor);
                                        %slumpa ut minorna
numgminor=numgrannminor(minefield);     %numminor är en matris där det i
                                        %varje ruta står hur många
                                        %grannminor det finns.


gmatris=grannmatris;                    %visar vad varje ruta har för
                                        %grannar
gmatris(storlek^2+1,storlek^2+1)=antminor;
                                        %sista raden i gmatris visar hur
                                        %många minor det totalt finns på
                                        %spelplanen
vilka=zeros(1,storlek^2+1);             %vilka ska vara en vektor där de
                                        %rutor som öppnats är ettor
vilka(1,storlek^2+1)=1;
first1=round(rand*(storlek-1))+1 ;      %väljer rad för första klicket
first2=round(rand*(storlek-1))+1 ;      %väljer kolumn för första klicket
first3=((first1-1)*(storlek))+first2;   %nr på rutan för första klicket
if minefield(first1,first2)==1          %om första klicket är på en mina
                                        %avbryt
   m=antminor;                          %och returnera antal minor
   return
else
   gmatris(first3,storlek^2+1)=numgminor(first1,first2);
                                        %sätt antalgrannminor på
                                        %rätt rad i gmatris


gmatris(:,first3)=0;
end
vilka(1,first3)=1;                      %addera den öppnade rutan till
                                        %vilka
tagna=find(vilka);                      %tagna visar siffror på de
                                        %öppnade rutorna
ejtagna=find(1-vilka);                  %ejtagna visar siffror på de ej
                                        %öppnade rutorna
vilka2=vilka;
```

```
while(antminor>0)                          %så länge det är kvar minor att
                                           %upptäcka
   sannolikhet=testbinary8(gmatris([tagna],:));
                                           %räknar ut minsannolikheten
                                           %för alla rutor
   sant=1;
   ruta=1;
   mina=0;

   for i=1:length(ejtagna)                 %loopa igen ejtagna-vektorn
      if ((sannolikhet(1,(ejtagna(i)))==1) & (mina==0))
                                           %om första hittade minan i
                                           %ejtagna som ej tidigare valts
         antminor=antminor-1;
         mina=1;

         ruta=ejtagna(i);                  %kom ihåg var minan låg
      end

      if ((sannolikhet(1,(ejtagna(i)))<=sant) & (mina==0))
                                           %välj den ruta med minst
                                           %minsannolikhet
         sant=sannolikhet(1,(ejtagna(i)));
         ruta=ejtagna(i);
      end
   end

   if mina==0                              %om ingen säkerställd mina bland
                                           %de oöppnade rutorna
      rad=(ceil(ruta/storlek));            %ta fram koordinater för den
                                           %valda rutan
      kol=(rem(ruta,storlek));
      if kol==0
         kol=storlek;
      end

      if minefield(rad,kol)==1             %om den valda rutan är minerad
         m=antminor;                       %antal minor kvar att röja
         return
      end
```

```
   end

   vilka(1,ruta)=1;                          %den valda rutan läggs på minnet
   if mina==0                                %om den valda rutan ej är en mina
      vilka2(1,ruta)=1;
      tagna=find(vilka2);
   end

   ejtagna=find(1-vilka);

   if mina==0
      gmatris(ruta,storlek^2+1)=numgminor(rad,kol);
                                             %sätt in information i gmatris
      gmatris(:,ruta)=0;                     %rutan ska ej vara med i
                                             %sannolikhetsberäkningarna mer
   end
end

m=antminor;                                  %antal minor kvar. Om 0 är spelet
                                             %klarat.
```

```
function y = testbinary(A);
%funktionen hittar alla binära lösningar till ekvationssystemet A


C=rref(A);                                   %utför Gauss Jordan elimination
                                             %på A
[m,n]=size(C);                               %m = antal rader, n = antal
                                             %kolumner
D=C;

for i=1:m                                    %hittar de första ettorna i resp
                                             %rad, dvs kollar vilka variabler
                                             %som är fria.
   j=1;
   while ((C(i,j)==0) & (j<n-1))
      j=j+1;
   end
```

```
   if C(i,j)==1
      B(j)=C(i,j);
      D(i,j)=0;                          %de första ettorna i resp rad
                                         %sätts till 0 i D.
   end


end
j=length(B);
while (n-1)~=size(B)
   B(j+1)=0;
   j=j+1;
end
B=1-B;                                   %B visar ettor för de fria
                                         %variablerna och nollor för de
                                         %beroende
E=D(:,n);                                %F ska göra om systemet så att de
                                         %beroende variablerna står
                                         %ensamma
E=E';
F=D(:,1:n-1);
F=(-1)*F;
F=F';
F=[E;F];
F=F';
H=find(B);                               %H visar på vilken plats i B de
                                         %fria variablerna finns
H1=find(1-B);

l2=length(H);                            %l2=antal fria variabler
l3=length(H1);
F2=zeros(n-1,n);
for i=1:l3
   F2(H1(i),:)=F(i,:);                   %F2 är som F men med de ber. var
                                         %på rätt rad
end


Add2=delaupp3(l2,l3,F,H1);               %Dela upp matrisen i oberoende
                                         %öar
j1=1;
```

```
koll=1;
sannolikhet=zeros(1,n-1);
while(koll<=length(find(Add2)))          %så länge det finns ober. Öar
                                         %kvar att räkna minsannolikheter
                                         %för.
   j2=1;
   ber=0;
   while Add2(j1)~=0
      ber(j2)=Add2(j1);
      j1=j1+1;
      koll=koll+1;
      j2=j2+1;
   end
   ber;


   F3=nyF2(ber,F2);
   h3=findfree2(ber,F3);

   G2=binmatris2(length(h3),n,h3);       %Nu vill jag få en matris G2, där
                                         %alla 0-1 kombinationer av de
                                         %fria variablerna finns med.

   t=F3*G2;                              %nu får jag alla lösningar på de
                                         %beroende variablerna där jag
                                         %testat alla 0-1 kombinationer av
                                         %de fria variablerna.

   Ny=G2(2:n,:);                         %ta bort hjälpraden med ettor.
   Ny=t+Ny;
   Ny=Ny';                               %för att bara få fram de binära
                                         %lösningarna multiplicerar jag
                                         %varje element i matrisen med sig
                                         %själv. De rader som ej ändras
                                         %får vara kvar i Ny3.
   Ny2=Ny.^2;
   [kk,ll]=size(Ny2);
   Ny3=zeros(1,ll);
   j=1;
   for i=1:2^length(h3)
      if Ny2(i,:)==Ny(i,:);
```

```
            Ny3(j,:)=Ny(i,:);
            j=j+1;
        end
    end
    Ny3;
    [q,p]=size(Ny3);                    %sannolikheterna kan nu räknas ut
                                        %genom att summera kolumnerna och
                                        %dividera med antalet lösningar
    if q==1
        kolsum=Ny3;
    else
        kolsum=sum(Ny3);
    end
    sannolikhet1=kolsum/q;
    sannolikhet=sannolikhet1+sannolikhet;
    j1=j1+1;
end
y=sannolikhet;
```