SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Optimiranje procesa automatske izrade tiskanih pločica

Josip Feliks

Voditelj: prof. dr. sc. Domagoj Jakobović

Zagreb, svibanj, 2014.

**Sadržaj**

[1. Uvod 1](#_Toc389510433)

[1.1 Opis stroja za postavljanje komponenata na tiskanu pločicu 1](#_Toc389510434)

[2. Definiranje optimizacijskog problema 3](#_Toc389510435)

[3. Potencijalni smjer rješavanja problema 4](#_Toc389510436)

[4. Rješenje 5](#_Toc389510437)

[4.1 Izrada simulatora 5](#_Toc389510438)

[4.2 Rezultati 5](#_Toc389510439)

[5. Zaključak 7](#_Toc389510440)

[6. Literatura 8](#_Toc389510441)

# Uvod

Tiskana pločica je pločica pomoću koje se elektroničke komponente međusobno povezuju. Postupak izrade tiskane pločice se kroz godine mijenjao. Prve tiskane pločice su imale samo jednu vodljivu stranu, tj. komponente su se nalazile s jedne strane pločice, a njihovi izvodi su prolazili kroz rupice koje su se bušile na tiskanoj pločici i povezivale se na drugoj strani. Kasnije su se radile tako da su obje strane pločice vodljive, tj. da se komponente mogu povezati preko obje strane. Tim procesom se smanjio potrebni prostor. S vremenom, kako je trebalo sve više komponenata smjestiti na što manji prostor, nastale su višeslojne pločice. To su u biti više međusobno povezanih jednoslojnih ili višeslojnih pločica.

U današnjoj industrijskoj proizvodnji se sve više primjenjuje tehnologija površinskog montiranja (engl. *Surface Mount Technology*). To je tehnologija kod koje se komponente montiraju na tiskanu pločicu tako da se pločica ne buši nego se izvodi komponente leme na istoj strani pločice na kojoj je i komponenta. Na taj način se štedi na prostoru jer izvodi iz komponenti mogu biti kraći. Naravno, i dalje postoje komponente koje nisu prikladne za površinsko montiranje pa se one i dalje postavljaju tehnologijom bušenja rupa kroz tiskanu pločicu.

Tehnologija površinskog montiranja je predstavljena 1960-ih, a u široku primjenu je ušla u kasnim 1980-im godinama. IBM je bila prva tvrtka koja ju je počela primjenjivati [2].

Neke od prednosti u odnosu na tehnologiju bušenja rupa:

• Manje komponente

• Manje rupa potrebno za izbušiti

• Brža izrada tiskanih pločica (neki strojevi su sposobni postaviti 136 000 komponenata u jednom satu)

• Bolje ponašanje prilikom vibracija

Nedostaci:

• Skuplji i teži popravak

• Nepogodna je za strujne krugove s visokim naponom

## Opis stroja za postavljanje komponenata na tiskanu pločicu

Za cijeli proces postavljanja potrebna su tri stroja. Prvi stroj nanosi lemnu pastu na tiskanu pločicu. Drugi stroj postavlja komponente na odgovarajuće pozicije na pločici dok treći stroj koristeći vrući zrak lemi komponente na pločici.

Fokus ovog seminara je na stroju koji postavlja komponente na tiskanu pločicu. On se sastoji od glave koja ima četiri nožice (engl. *spindle*) i trake (engl. *feeder*) s kojih glava skida komponente koje je potrebno postaviti. Ovisno o veličini, moguće je da stroj ima više glava i nožica te više traka s komponentama. Svaka traka je podijeljena na slotove na kojima se nalaze komponente. Primjer jednog SMT stroja za postavljanje komponenata prikazan je na slici 1.



Slika 1 (preuzeta s [1]): SMT stroj za postavljanje komponenata

U ovom seminaru će naglasak biti na optimizaciji redoslijeda uzimanja komponenata s traka. U 2. poglavlju će se definirati optimizacijski problem za koji će se u 3. poglavlju pokušati naći pogodan smjer rješavanja, koje tehnike i alate koristiti itd. Nakon što se odredi način na koji će se problem pokušati riješiti, u 4. poglavlju predstoji izrada simulatora kojim bi se pronašlo najbolje moguće rješenje. Kao zaključak (6. poglavlje) navodi se je li odabrani smjer dao zadovoljavajuće rezultate, ima li smisla nastaviti u istom smjeru ili je potrebno nešto mijenjati.

# Definiranje optimizacijskog problema

SMT stroj za postavljanje komponenata može imati više glava koje uzimaju i postavljaju komponente na pločicu. Međutim, radi jednostavnosti, u ovom radu smatra se da stroj ima jednu glavu na kojoj postoje četiri nožice. Svaka od tih nožica je numerirana i postoje koordinate za nju. Stroj radi na taj način da sa trake uzme 4 komponente, svaka komponenta na jednu nožicu, i zatim ih postavi na pločicu. Kao što je napisano u uvodu, ovdje je naglasak na optimizaciji redoslijeda uzimanja tih komponenata. Nakon što je stroj uzeo četiri komponente, glava stroja mora otići u prije definirani položaj koji postaje početni položaj prilikom postavljanja. Nakon što stroj postavi sve četiri komponente, položaj u kojem se trenutno nalazi postaje početni položaj za ponovno uzimanje komponenata s traka. Cilj je da vrijeme potrebno za cijeli taj postupak bude najmanje moguće i računa se po formuli 1.

$$t\_{uk}= t\_{1}+t\_{2}+t\_{3}+t\_{4}+t\_{5}$$

$t\_{i}= \frac{s}{v}$ **,** $1\leq i\leq 5$

Formula 1: vrijeme koje je potrebno minimizirati

Vrijeme $t\_{1}$ je potrebno da glava stroja iz početnog položaja dođe do prve komponente na traci i uzme ju, $t\_{2}$ da glava stroja iz trenutnog položaja dođe do druge komponente i tako dalje sve do $t\_{5}$ koje označava vrijeme potrebno da se glava iz trenutnog položaja vrati u početni položaj.$ v$ je brzina glave stroja i to je konstanta koja se iščitava iz konfiguracije stroja.

Nakon što glava uzme komponentu s trake, potrebno je neko vrijeme da nova komponenta na traci dođe u početni položaj. U nekim situacijama isplativije je proći neki put i uzeti drugu komponentu pa se vratiti u prijašnji položaj i opet uzeti onu komponentu koja je uzeta prva jer vrijeme postavljanja komponente u početni položaj može trajati duže od pomicanja glave na neku drugu lokaciju.

# Potencijalni smjer rješavanja problema

Kao potencijalni smjerovi rješavanja nameću se genetski algoritam i algoritam iscrpne pretrage. Genetski algoritmi su pogodni kad je potrebno ispitati veliki broj permutacija. Za manji broj mogućih permutacija isplati se koristiti algoritam iscrpne pretrage jer, za razliku od genetskog algoritma, on sigurno nalazi optimalno rješenje, a razlika u vremenima pronalaska rješenja je mala. Ako problem zahtjeva ispitivanje velikog broja permutacija, i uz to je bitno i vrijeme potrebno za pronalazak rješenja, isplati se koristiti genetske algoritme.

Pretpostavka je da su sve komponente jednakih dimenzija i da svaka nožica može uzeti bilo koju komponentu. Također, postoji jedna glava s četiri komponente (svaka ima svoje koordinate) te jedna traka s komponentama. Traka je podijeljena na više vertikalnih dijelova (slotova) na kojima se nalaze komponente. Svaki slot trake ima svoje koordinate tako da se može izračunati put od glave do određene komponente.

S obzirom na to da su neke komponente „više“ od drugih, postoje neka ograničenja kojih se treba pridržavati, npr. prvo je potrebno postaviti „niže“ komponente, a nakon toga „više“. Iz toga razloga već postoji redoslijed komponenti koji je podijeljen u manje skupine od po četiri komponente. Simulator mora za svaku skupinu od četiri komponente pronaći put za koji će biti potrebno najkraće vrijeme da glava „pokupi“ sve komponente.

Broj mogućih permutacija rješenja za svaku skupinu od četiri komponente je 576. Razlog je taj što za prvu komponentu postoje četiri slobodne nožice koje ju mogu uzeti. Drugu komponentu mogu uzeti preostale tri slobodne nožice. Treću komponentu mogu uzeti preostale dvije nožice, a zadnju komponentu uzima jedina slobodna nožica. Broj mogući permutacija za taj postupak je 4! (4 \* 3 \* 2 \* 1). Za svaku od tih permutacija potrebno je odrediti redoslijed uzimanja. Nije nužno da prvo komponentu uzima prva nožica. Iz toga razloga, također postoji 4! mogućih permutacija. Ukupno postoji 576 (4! \* 4!) mogućih rješenja. Taj broj nije veliki i zato se isplati koristiti algoritam iscrpne pretrage.

Skupinu komponenti potrebno je učitati iz tekstualne datoteke.

.

# Rješenje

## Izrada simulatora

Za implementaciju simulatora koristit će se programski jezik *Java*. Svi potrebni podaci koji su skloni promjenama (npr. broj glava stroja, razmak između nožica na glavi, broj komponenata koje je potrebno postaviti, brzina glave stroja, dimenzije, itd.) smješteni su u *.xml* datoteci koju simulator u početku čita i na temelju tih podataka pokreće simulaciju.

Tijek rada simulatora:

1. Učitati podatke iz *.xml* datoteke koja se odnosi na stroj
2. Učitati listu komponenti koje je potrebno postaviti
3. Pronaći najbolji mogući redoslijed uzimanja komponenata

## Rezultati

Ovisno o zadanim specifikacijama i komponentama koje je potrebno postaviti, rješenja se razlikuju. U testnom primjeru bilo je potrebno odrediti najbolju permutaciju za 13 grupa. Vrijeme izvođenja bilo je vrlo kratko, oko jedne sekunde, iz čega se vidi da je algoritam iscrpne pretrage bio dobar odabir kao način rješavanja.

Za svaku grupu od po četiri komponente izračunato je vrijeme njihovog uzimanja sa slotova u početnom redoslijedu koji je naveden u ulaznoj datoteci, i nakon provedenog algoritma iscrpne pretrage. Tablica 1 prikazuje rezultate. Osim za svaku grupu, izračunato je i ukupno vrijeme. Za ove ulazne podatke, koristeći redoslijed koji je zadan na početku, vrijeme potrebno da se sve komponente uzmu je 42.987 sekundi dok nakon algoritma iscrpne pretrage to vrijeme iznosi 9.2 sekunde. Vidi se da je ukupno vrijeme nakon algoritma kraće za 78.6% što je veliko poboljšanje.

Tablica 1: Postignuto poboljšanje

|  |  |  |
| --- | --- | --- |
|  | SKUPINE OD PO 4 KOMPONENTE |  |
| 1. | 2. | 3. | 4. | 5. | 6. | 7. | 8. | 9. | 10. | 11. | 12. | 13. | Ukupno |
| **Kraće vrijeme****(u %)** | 85 | 95 | 72.1 | 45.18 | 85.66 | 69.37 | 82.22 | 66.6 | 58.69 | 78.86 | 89.44 | 90.24 | 85 | 78.6 |

U slučaju da je potrebno pronaći najbolje rješenje koje uključuje i postavljanje komponenata, broj mogućih permutacija bi narastao i pitanje je bi li se isplatilo koristiti algoritam iscrpne pretrage. Dodatnu kompleksnost bi zadala potreba za određivanjem skupina od četiri komponente. U tom slučaju se definitivno ne isplati koristiti algoritam iscrpne pretrage već genetski algoritam.

# Zaključak

Nakon dobivenih rezultata može se zaključiti da je za ovaj problem algoritam iscrpne pretrage bio prikladan. Ovo je samo prvi korak kod optimizacije postavljanja elemenata na pločicu s obzirom da je ovdje prikazano samo uzimanje komponenata, a ne i njihovo postavljanje. Vrlo vjerojatno je da bi se redoslijed uzimanja promijenio kad bi se realiziralo i postavljanje komponenata. Razlog je taj što se u ovom slučaju, nakon što se odaberu četiri komponente, ode uvijek u isti položaj koji postaje početni za sljedeću iteraciju. Komponente se ne postavljaju uvijek na isto mjesto pa bi se vrijeme potrebno da se „uzme“ prva komponenta u svakom slučaju razlikovalo. Stoga, da bi ovaj problem bio kompletno riješen, potrebno je realizirati i postavljanje komponenata. Za to bi, kao što je već navedeno isplativije bilo koristiti genetske algoritme.

# Literatura

[1] <http://en.wikipedia.org/wiki/SMT_placement_equipment>

[2] <http://en.wikipedia.org/wiki/Surface-mount_technology>