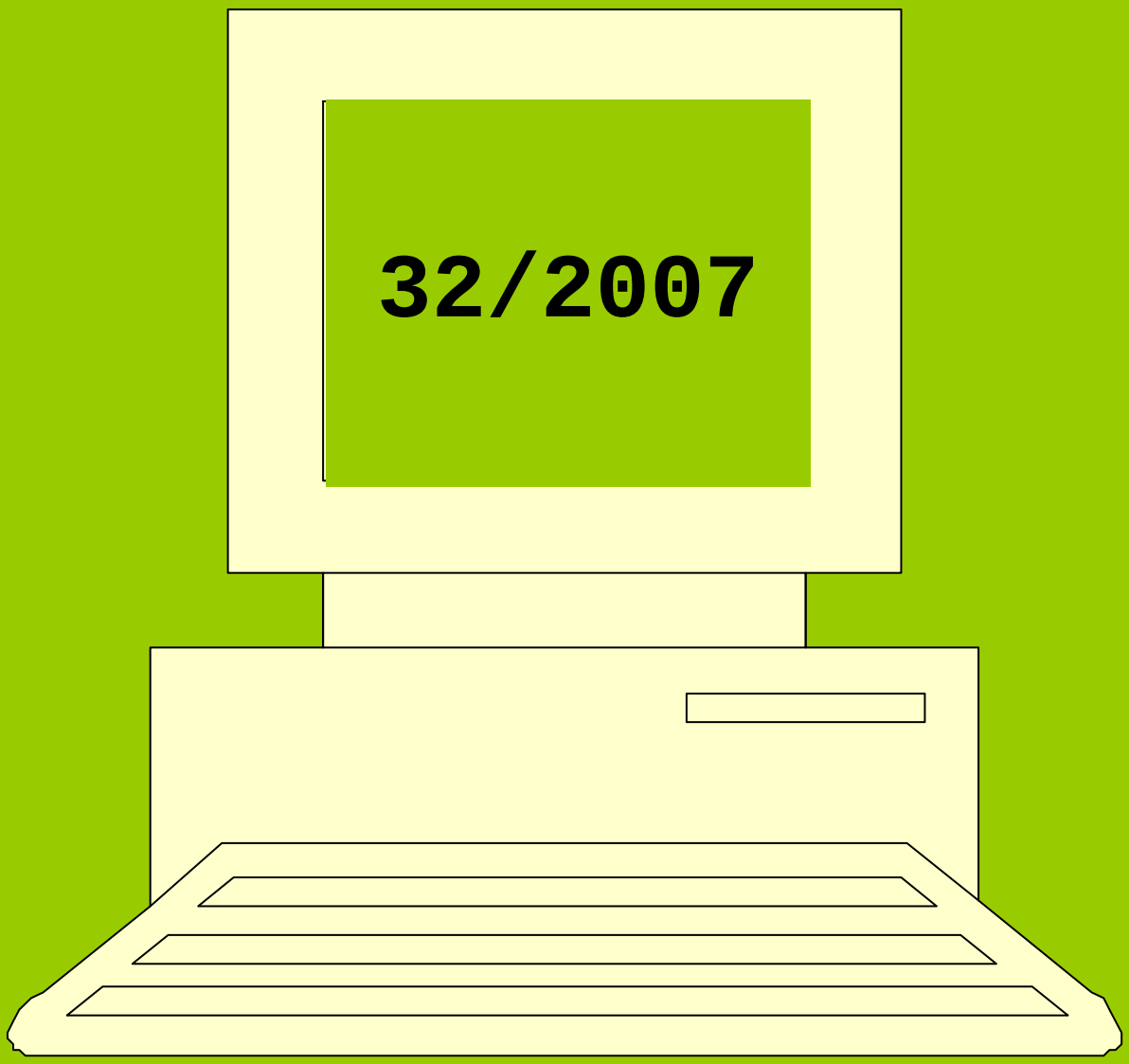
INFORMATIKA
V ŠKOLE



Informatika v škole č. 32 2007

ISSN 1335-616X

Informačné periodikum

O teoretických, metodických otázkach a skúsenostiach z praxe pri uplatňovaní informatiky a výpočtovej techniky v základných a stredných školách

Výkonná redaktorka: Ing. Alžbeta MEGOVÁ

Vydáva

Ústav informácií a prognóz školstva v Bratislave

Adresa redakcie: Ústav informácií a prognóz školstva
Staré grunty 52
842 44 Bratislava
e-mail: megova@uips.sk

OBSAH

VYUŽITIE NEURÓNOVÝCH SIETÍ V BEŽNÝCH ALGORITMOCH

Kľúčové slová: *neurónová sieť, adaptácia, genetický algoritmus, Fourierova transformácia*

Neurónové siete je možné využiť ako súčasť klasických algoritmov. Pyramídová topológia dopredných sietí vybavená genetickým adaptačným algoritmom môže danú implementáciu dopĺňať učiacim sa systémom, ktorý je schopný vykonávať rozhodovania v oblastiach matematickej lingvistiky, rozpoznávania objektov, filtrácie nežiadúcich informácií, prepínania smerov dát na uzloch internetu.

1. Neurón

Zvykli sme si prijímať zázraky, ktoré nás obklopujú, s úplnou samozrejmosťou.

Telá vyšších biologických organizmov obsahujú nervovú sústavu, ktorej základnou logickou jednotkou je bunka nazývaná neurón. Neurón pozostáva vo všeobecnosti z tela bunky a z dvoch druhov výbežkov. Kratšie, dostredivé výbežky, nazývame dendrity [1], dlhší, odstredivý výbežok, nazývame neurit alebo axón. Dostredivé, vstupné výbežky majú dĺžku niekoľko milimetrov, odstredivý, výstupný výbežok dosahuje dĺžku až niekoľkých metrov. Je zaujímavé uvedomiť si, že ak sa mechanicky preruší nerv v tele živočícha, preruší sa zväzok axónov. Inými slovami sa veľkému množstvu neurónov odsekne veľká časť ich najdlhšieho výbežku – axónu. Hojenie takéhoto zranenia je v zásade neuspokojivé a pomalé.

Dendrity a axón sú akýmisi biologickými vodičmi elektrického náboja, ktorý sa prenáša repolarizáciou elektrického náboja na bunečnej membráne. Neurón teda prenáša a spracúva elektrické impulzy. Axóny jedných neurónov sú pripojené na dendrity iných neurónov, čím je vytvorená nervová sústava organizmu, z hľadiska informatiky môžeme rozprávať o neurónovej sieti. Je veľmi zaujímavé uvedomiť si, že sa rozhodujeme a premýšľame pomocou bioelektronického informačného systému, ktorý je schopný učiť sa, spracúvať nové i staré informácie.

Pokúsme sa teraz zamyslieť nad nervovou sústavou organizmov v takom zmysle, ktorý bude oddeľovať logiku rozhodovania od jej formalizácie, popisu, od porozumeniu logických podstát.

Letec sa v škole učí rozumieť rámcovo viacerým oblastiam, ktoré sú pre neho teoreticky potrebné. Medzi takéto znalosti patrí napr. mechanika, elektronika, navigácia, znalosti z oblasti meteorológie, obsluha konkrétnych lietadiel. Po ukončení štúdia a praktických cvičení je k dispozícii vzdelaný človek, schopný bezpečne riadiť lietadlo.

Pokúsme sa sledovať v prírode alebo v dokumentárnom prírodopisnom filme techniku letu niektorých vtákov. Automaticky sa ponúkne otázka: Kde sa to naučili? Ich mozog je oveľa jednoduchší než ľudský mozog, avšak ich letové formácie a techniky sa často vyrovnajú formáciám vojenského letectva.

Alebo sa môžeme opýtať: Kedy sme videli muchu, ktorá by mala problémy s lietaním, pristávaním alebo navigáciou? Mucha má len veľmi jednoduchú neurónovú sieť dokonca aj v porovnaní s vtáčou.

Môžeme si teda položiť ďalšiu otázku, ktorá sa už bude týkať výlučne informatiky: Koľko výpočtového času a zložitosti mozgu vyžaduje formalizácia, porozumenie problému a koľko samotná zručnosť so skúsenosťami, ale bez potreby porozumenia a formalizácie?

Neurónová sieť je výpočtový systém, ktorý si svoje znalosti ukladá do nastavenia váh vstupov jednotlivých neurónov [2,3,4,5]. Daná znalosť – postupnosť váh – je nazývaná konfiguráciou neurónovej siete. Ak je sieť naučená – adaptovaná, jej konfigurácia zabezpečuje správny výpočet, rozhodovanie. Súvislosti medzi vnútornými hodnotami konfigurácie sú však pritom neznáme. Konfigurácia je postupnosťou čísel, medzi ktorými sa nedajú nájsť pravidlá ako napríklad: Ak klesáš, zamávaj krídlami. Jednotlivé informácie v neurónovej sieti nie sú ukladané na konkrétne miesta! Sú rozkladané do množstva hodnôt danej konfigurácie, takže je spätne prakticky nemožné získavať porozumenie, prečo daný neurónový systém reagoval tak alebo inak. Takáto forma ukladanie informácií má dve zaujímavé výhody:

1. Každá informácia je rozložená do priestoru celej konfigurácie. Porucha v konfiguračnej pamäti preto nespôsobí úplné zlyhanie, spôsobí len stratu istej miery presnosti rozhodovania.
2. Informácie, ktoré sú ukladané bez formalizácie a súvislostí, vyžadujú omnoho menej pamäte a výpočtového času. Obyčajná mucha je výborným letcom. Sťahovavé vtáky nepoužívajú mapy, ale aj tak sa bezpečne vracajú na svoje letoviská.

Pokúsme sa teraz túto druhú vlastnosť neurónových systémov využiť v rámci prvkov umelej inteligencie, ktoré budú implementované do klasických algoritmov.

2. Kompresia dát so stratou informácie

Uvažujme o topológii neurónovej sieti, v rámci ktorej budú neuróny usporiadané do jednotlivých vrstiev. Vstupná vrstva neurónov – receptorov bude najširšia, bude mať najviac prvkov. Každá ďalšia vrstva bude mať R krát menej neurónov než je na predchádzajúcej vrstve, kde 1<R<2. Poslednú vrstvu budeme nazývať výstupnou vrstvou neurónov – efektorov.

Výstupy neurónov z N-tej vrstvy budú pripojené na vstupy neurónov z N+K-tej vrstvy pre k>=1. Takúto topológiu siete budeme nazývať doprednou pyramídovou topológiou.

Šírka vstupu neurónovej siete je daná počtom receptorov. Vstupnou hodnotou siete je teda vektor, ktorého rozmer musí korešpondovať so šírkou vstupu siete. Takéto vektory budeme ďalej nazývať vstupné vektory.

Podobne počet efektorov určuje rozmer výstupných vektorov [1, 2, 3, 4, 5].

Neurónová sieť vyhodnocuje vstupný vektor v tzv. aktívnom režime. Pretože rozprávame o doprednej sieti s pyramídovou topológiou takou, že výstupný vektor má menší rozmer než ten vstupný, pri vyhodnocovaní vstupu dochádza k strate informácií.

Strata informácií môže byť deštruktívna, ale môže byť aj cielená, dôsledne riadená výpočtovým systémom. V oblasti vyhodnocovania dát pomocou neurónovej siete musí mať jej konfigurácia také hodnoty, aby sa strácali nepodstatné alebo rušivé dáta. Sieť musí byť teda správne naučená, adaptovaná.

Nepodstatné dáta sa môžu strácať napríklad pri kompresii zvuku alebo obrazu – rušivé dáta sa odstraňujú pri filtrácii šumu zo zvukového signálu. Teraz sa však zamyslíme nad úplne odlišnou filtráciou údajov.

Keď sedí mucha na stole v kuchyni a blíži sa k nej človek so zdvihnutou rukou, mucha nerieši problém, či ten človek je cvičený karatista alebo len žena v domácnosti. Podobne ani neodhaduje, aký rýchly bude daný človek vzhľadom na jeho nadváhu. Receptory mozaikového mušieho videnia zaznamenajú zmenu obrazu – široký vektor. Jej neurónová sieť odovzdá skalár – akútne ohrozenie. Mucha potom bez uvažovania reaguje.

Podobne, pudovo reagujú aj zvieratá bez potreby formalizácie vstupných údajov. Je zaujímavé uvedomiť si, že vnímaný obraz, zvuk a možno aj s nejakou hmatovou zložkou v sebe nesie ukrytú informáciu, ktorá poukazuje na bezpečie alebo ohrozenie. Skúsenosťami natrénovaná neurónová sieť vyhodnotí vstupné údaje bleskovo, a to aj v prípade muchy, ktorá má nepomerne menej neurónov než vyššie živočíchy.

Takto si môžeme položiť otázku, aké informácie chceme filtráciou odstrániť a o aké nám ide? Cieleným natrénovaním neurónovej siete dostaneme algoritmický filter, ktorý bude pracovať v aktívnom režime rýchlo, svojou funkciou môže dopĺňať klasický algoritmus v mieste, kde sa vyhodnocovanie vstupných údajov formalizuje veľmi obtiažne. Správna funkcie siete získaná jej tréningom bude uložená v „tajomnom orákulu“ – v nastavení váhových vektorov jednotlivých vstupov neurónov, teda v danej konfigurácii siete.

3. Algoritmické spracovanie vstupných dát

Neurón je možné modelovať v oblasti informatiky napríklad pomocou datových štruktúr ako záznam alebo objekt. Vstupy neurónu – dendrity – vytvárajú vektor, z hľadiska informatiky teda pole reálnych hodnôt. Každá hodnota musí mať svoju vstupnú váhu. Váhový vektor je teda tiež realizovaný pomocou poľa reálnych hodnôt váh jednotlivých vstupov. Každý neurón má svoju vnútornú reálnu hodnotu – potenciál neurónu a výstupnú hodnotu, ktorá vyjadruje aktivitu axónu.

K daným dátovým štruktúram musí byť priradená funkcia na výpočet potenciálu neurónu, vstupom tejto funkcie je vstupný a váhový vektor. Druhou funkciou je tzv. aktivačná funkcia, ktorej vstupom je potenciál neurónu a výstupom je vlastná výstupná hodnota neurónu.

Poprepájaním neurónov do siete – synaptické prepojenia – dostaneme konkrétnu topológiu, ktorá môže byť navrhnutá tak, aby daná neurónová sieť spĺňala požadované kritériá.

Interfejsom medzi klasickým algoritmom a neurónovou sieťou je vstupná vrstva receptorov a výstupná vrstva efektorov. Neurónová sieť musí dostať na svoj vstup vektor – teda pole reálnych hodnôt. Výstupom je opäť pole reálnych hodnôt.

Sieť je možné natrénovať tak, aby výstupný vektor reprezentoval také hodnoty, ktoré sa v ďalšom algoritmickom spracovaní budú požadovať. Týmito hodnotami môžu byť napríklad:

- Nula alebo jedna – podnet na vykonanie ďalšej akcie.
- Reálne číslo – miera následnej aktivity.
- Položky výstupného vektoru môžu byť konkrétnymi mierami výslednej entity, ako napríklad:

1. – O dve hodiny,

2. – bude pršať,

3. – teplota klesne na N stupňov.

O niečo zložitejším problémom je príprava vstupných údajov tak, aby boli neurónovej sieti reprezentované vždy ako postupnosť vektorov.

Majme napríklad postupnosť vzoriek zvuku, ktoré sú uchované v pamäti pomocou pulznej kódovej modulácie [1, 6, 7, 8]. Pri digitálnom spracúvaní zvuku je nutné prihliadať na uchovaný signál v zmysle jeho krátkodobých mikrosegmentových charakteristík [1, 6]. Jednotlivé mikrosegmenty signálu je možné previesť pomocou rýchlej Fourierovej transformácie na vektory – príznaky mikrosegmentov. Zvuk je takto možné spracovávať ako postupnosť príznakov – vektorov. Bloky zvuku v takejto forme je možné teda spracovávať aj pomocou neurónovej siete.

Raster grafických informácií je možné previesť na postupnosť lokálnych matíc pixelov. Pomocou Fouriera je opäť možné previesť tieto matice na matice reálnych čísel, ktoré sa dajú reprezentovať po riadkoch ako vektor a použiť ich tak opäť ako vstup neurónovej siete.

Tým, že sa vstupné aj výstupné údaje môžu reprezentovať vo vektorovej forme, je možné použiť neurónový systém ako súčasť klasických algoritmov. Algoritmy takto môžu byť rozšírené o prvky umelej inteligencie, schopné učiť sa a reagovať na základe nových poznatkov bez nutnosti ich formalizácie.

4. Adaptácia siete

Veľmi dôležitou súčasťou neurónových systémov je adaptačný proces, v rámci ktorého sa neurónová sieť učí reagovať na nové, nepoznané podnety. Proces adaptácie je v princípe oddelený od práce siete v aktívnom režime, v ktorom prebieha samotné rozhodovanie – funkcia siete v rámci klasického algoritmu.

Adaptačný proces je pomerne zdĺhavý, lebo sa jedná vo svojej podstate o optimalizačný algoritmus. Adaptácia preto musí prebiehať mimo práce siete v aktívnom režime alebo musí mať sieť k dispozícii softvérové prostriedky, aby mohla adaptácia bežať na pozadí v čase, keď je počítač nevyužívaný.

Aby bol proces adaptácie použiteľný, musí spĺňať dve dôležité kritériá:

- Musí pracovať dostatočne rýchlo.
- Nesmie uviaznuť v lokálnom extréme.

Genetický algoritmus svojou funkciou zabezpečuje, že konvergencia procesu adaptácie smeruje ku globálnemu riešeniu. Genetický algoritmus je pomerne rýchly proces založený na pravdepodobnostnom princípe. Ďalej uvediem jedno vylepšenie genetického algoritmu, ktoré budem skrátene nazývať GAS – genetický algoritmus so šľachtením.

GAS používa pri svojej funkcii tri genetické princípy [1, 2, 9, 10]:

1. náhodný výber

2. kríženie

3. mutácia

Tieto tri princípy sú v rámci GAS realizované pomocou operátorov selekcie, kríženia a mutácie. GAS rozširuje túto trojicu operátorov ešte o jeden - operátor šľachtenia.

Ponúkajú sa nám otázky: Z čoho ideme vyberať? Čo ideme šľachtiť? Čo ideme krížiť a mutovať?

Znalosťou neurónovej siete je daná konfigurácia. Majme teraz celú populáciu konfigurácií. Jedna konfigurácia bude v našom ponímaní jeden genetický reťazec, ktorý budeme „lámať“ a kombinovať s časťami iných genetických reťazcov tak, aby nakoniec vznikali genetické jedince, reťazce, konfigurácie, ktoré budú pre danú neurónovú sieť v aktívnom režime lepšou znalosťou. Aby mohol takýto proces fungovať, musíme korektne uplatniť všetky genetické princípy – vyberať jedincov s populácie, krížiť ich, mutovať, šľachtiť jedincov s najlepšími vlastnosťami a vytvárať tak v rámci populácie novú generáciu konfigurácií.

GAS vnáša do štandardného prevedenia genetického algoritmu tri nové prvky [1, 2, 9, 10]:

1. Pracuje nad jedinou populáciou konfigurácií

2. Najhoršie konfigurácie eliminuje

3. Obsahuje štvrtý operátor - šľachtenie

4.1 Populácia viacerých generácií

Biologické organizmy špecifikovaného druhu v prírode existujú v rámci jednej populácie. V tejto populácii môžeme nájsť starších, mladších jedincov, zdravých i chorých... Medzi kvalitatívnou rozmanitosťou jedincov prebieha prirodzená selekcia a kríženie.

Majme GAS s jedinou populáciou konfigurácií. Každej konfigurácii je priradená celočíselná hodnota bez znamienka – vek danej konfigurácii. Hodnota 0=najmladší, maximum=najstarší, zväčšenie premennej veku o jedna budeme nazývať zostarnutím o jeden rok. Počas behu algoritmu prebieha proces starnutia konfigurácií nasledovne:

a) Pri krížení bežného otca a matky títo rodičia zostarnú o jeden rok.

b) Pri krížení elitného otca a matky títo rodičia X-krát nezostarnú – X je riadiaca hodnota operátora šľachtenia.

c) Pri každom krížení sa priradí synovi a dcére najmladší vek.

Vek danej konfigurácii je zároveň jednou z váh pri váženej selekcii najlepších. Inými slovami aj tá najkvalitnejšia konfigurácia, keď zostarne, začína byť zlou, na základe čoho sa stáva relevantným kandidátom na elimináciu.

Z vyššie uvedeného textu vyplýva, že keby proces starnutia prebiehal príliš rýchlo, v populácii by sa strácali kvalitné konfigurácie. Ak je ale proces starnutia dostatočne pomalý, likviduje v populácii konfigurácie s vysokou mierou morálnej zastaralosti.

4.2 Výber dvojíc a kríženie s elimináciou

Pri štandardnom genetickom algoritme selektor vyberie vo váženom náhodnom zmysle najlepšiu konfiguráciu. Dvojnásobným použitím selekcie sa vyberú rodičia na kríženie.

V GAS sa tiež selektor použije dvakrát. Každé použitie selektora však vyberie dvojicu – vo váženom náhodnom zmysle jednu najlepšiu a jednu najhoršiu konfiguráciu. Dve najlepšie sa stanú rodičmi a dve najhoršie sa eliminujú – prepíšu sa deťmi daných rodičov.

Pri jednom priechode populáciou selektor dvojíc v GAS rozdelí populáciu na dve polovice:

- najmladšia generácia detí,
- starší i mladší rodičia.

V ďalšom kroku GAS prechádza opäť celou populáciou. Konvergenčnému kroku procesu GAS rozumieme, keď selektor vyberie ako rodiča aspoň jedno dieťa s vekom rovným nule.

Tým, že selektor GAS rozdeľuje populáciu na generáciu detí a generáciu starších rodičov, automaticky potláča lokálne divergencie, čím sa celkový proces adaptácie urýchli.

Keď pozorujeme zvieratá žijúce vo voľnej prírode, niekedy nás možno zarazí, aké sú zdravé, silné. Jeden zo surových, bezcitných prírodných mechanizmov, ktorý zabezpečuje silu a zdravie zvierat, je prirodzená schopnosť dravcov uloviť najslabšie, choré alebo staré jedince. V GAS túto funkciu modeluje selektor dvojíc a kríženie s elimináciou.

4.3 Genetický algoritmus so šľachtením

Do prirodzeného vývoja populácie je možné zasahovať istou formou uvedomelého prístupu – šľachtením.

Je prirodzené predpokladať, že najlepšie konfigurácie v sebe obsahujú najkvalitnejší genetický materiál. Stanovme teraz dve čísla:

SX – šírka šľachtenia

HX – hĺbka šľachtenia

Číslo SX určuje počet najlepších konfigurácií, ktoré budú podliehať šľachteniu. Číslo HX určuje, koľkokrát sa bude daná konfigurácia šľachtiť.

Pri operácii kríženia v GAS rodičia vždy zostarnú o jeden rok. Starnutie sa však netýka elitných konfigurácií, teda znalostí určených na šľachtenie. Po krížení sa navyše tieto konfigurácie neoznačia ako použité v zmysle priechodu algoritmu populáciou. Inými slovami sa elitné konfigurácie poskytujú operátoru kríženia viackrát, čím sa kvalitatívne vylepšuje genetický materiál novej generácie konfigurácií.

Aby priniesol operátor šľachtenia požadované zrýchlenie adaptácie, je potrebné správne nastaviť hodnoty SX a HX, ktoré by mali byť veľmi malé – približne do piatich percent veľkosti populácie.

Štandardný genetický algoritmus a aj GAS používajú pri svojom behu niekoľko konštánt, ktoré budeme nazývať riadiacimi vektormi GA alebo GAS. Kvôli porovnaniu sú nižšie uvedené obidva tieto riadiace vektory.

Riadiaci vektor GA.

1. veľkosť jednej populácie,
2. váha náhodného výberu najlepších konfigurácií,
3. percento kopírovaných najlepších konfigurácií,
4. počet bodov kríženia,
5. pomer početností kríženia a mutácie,
6. interval generátora reálnych náhodných hodnôt,
7. hranica maximálnej prípustnej chyby siete.

Riadiaci vektor GAS.

1. veľkosť populácie,
2. váha náhodného výberu najlepších a najhorších,
3. rýchlosť procesu starnutia,
4. počet bodov kríženia,
5. pomer početností kríženia a mutácie,
6. interval generátora reálnych náhodných hodnôt,
7. hranica maximálnej prípustnej chyby siete,
8. šírka operácie šľachtenia.
9. Hĺbka operácie šľachtenia

Vyššie uvedené riadiace vektory sa principiálne líšia v tretej, ôsmej a deviatej dimenzii.

Na záver tejto podkapitoly si ešte uvedieme schematický algoritmus GAS.

VAR otec,matka,syn,dcéra:Konfigurácie; i:INTEGER; min:REAL; BEGIN inicializácia FOR i:=1 TO N DO BEGIN otec.gén:=RANDOM(konfigurácia); otec.chyba:=účelová(otec.gén); otec.vek:=najmladší; vlož otca do populácie END; min:=maximum; REPEAT proces minimalizácie REPEAT priechod populáciou vyber najlepšieho otca a najhoršieho syna; vyber najlepšiu matku a najhoršiu dcéru; skríž otca a matku do syna a dcéry; eliminácia mutuj syna; mutuj dcéru; syn.vek:=najmladší; dcéra.vek:=najmladšia; šľachti otca; realizuje aj starnutie šľachti matku; realizuje aj starnutie syn.chyba:=účelová(syn.gén); IF syn.chyba < min THEN min:=syn.chyba; dcéra.chyba:=účelová(dcéra.gén); IF dcéra.chyba < min THEN min:=dcéra.chyba UNTIL priechod populáciou ukončený UNTIL min<koncové kritérium END.

5 Aplikácie neurovýpočtov

Na Ústave vedy a výskumu UMB v Banskej Bystrici je vyvíjaný špeciálny systém, ktorý by mal slúžiť na sprístupňovanie informácií nevidiacim používateľom osobných počítačov. Súčasťou vyvíjaného systému sú aj moduly s prvkami umelej inteligencie.

Keďže na tomto pracovisku je zamestnaný nevidiaci vedecký pracovník, špeciálny systém musí umožňovať nevidiacemu používateľovi aj prácu na úrovni vedeckej informatiky, a to konkrétne v oblasti vývoja neurónových sietí a tiež aj v oblasti syntézy a rozpoznávania plynulej reči.

Špecializovaný systém musí teda poskytovať nevidiacemu pracovníkovi adekvátne prostriedky:

- Rozhranie na definíciu topológie sietí bez vyžadovania grafických informácií.
- Musí poskytnúť primerané vývojové prostredie na testovanie sietí.
- Musí obsahovať prostredie na návrh a vytváranie hlasovej databázy pre vývoj syntetizéra.

Tým, že sa s vyvíjanými aplikáciami zároveň pracuje, prebieha automaticky testovanie behu aplikácií. Na záver tohoto článku budú ešte uvedené niektoré teoretické využitia prvkov umelej inteligencie v rámci bežných algoritmov, ktoré by pri vhodnom naprogramovaní aplikácií mohli zaujímavo pomôcť hendikepovaným ľuďom.

5.1 Kategorizačná sieť

V kapitole tri tohoto článku je uvedená zmienka o konverzii grafických informácií na vektory, ktoré môžu byť potom vstupom neurónovej siete. Ak máme správne natrénovanú doprednú pyramídovú neurónovú sieť, ktorej vstupom budú grafické obrázky, jej výstupné vektory môžu hodnotiť kategórie daných obrázkov. Jednotlivé skaláre daných vektorov môžu vyjadrovať početnosť danej kategórie ako napríklad:

1. Ľudia
2. Autá
3. Príroda
4. Zvieratá...

Obrázok môže byť takto kategorizovaný napríklad nasledovne: Dvaja ľudia, príroda a zviera. Malý počet kategórií nevyžaduje veľký rozsah siete. Topológiu je možné rozširovať, čím sa do výstupu pridajú ďalšie rozmery. Prakticky to znamená, že zviera už nemusí byť pomenované všeobecne, ale odovzdá sa napríklad jeho druh – mačka, pes.

Neurónový systém takto dokáže triediť obrázky. V aplikácii na využitie pre nevidiacich môže počítač hendikepovanému používateľovi povedať syntetickým hlasom, čo je zobrazené, v zmysle nejakého rámca – kategórie, na obrázku.

Dostatočne rozsiahla sieť s dosť širokým výstupným vektorom, ktorá by navyše prešla zložitým adaptačným vývojom, môže hodnotiť grafické informácie na úrovni, ktorá sa dá označiť ako rozpoznávanie obrazu.

5.2 Výslovnosť slov vzhľadom na kontext

V slovenskom jazyku existujú slová, ktoré sa musia vyslovovať rôznym spôsobom vzhľadom na kontext vety:

Videli sme KRÁSNE pohorie.

Bolo nám KRÁSNE, lebo sa o nás dobre starali.

VLÁDNI predstavitelia tam už neboli.

VLÁDNI si tu sám, keď si taký nespravodlivý!

Slová „krásne“ a „vládni“ majú vo vyššie uvedených vetách rovnaký zápis, ale rozdielnu výslovnosť. Zmäkčenie na konci slova nastane až na základe rozhodovania vyplývajúceho z kontextu daného slova.

Takéto rozhodovanie popri produkcii softvérového syntetizéra môže zabezpečovať neurónová sieť v aktívnom režime. Samotná produkcia reči pritom môže prebiehať v klasicky ponímanej syntéze v časovej oblasti [1, 2].

5.3 Smerovanie a filtrovanie dát

Natrénované prvky neurónových sietí môžu byť implementované do algoritmov, ktoré musia rýchlo voliť smer toku daných dát. Napríklad v oblasti filtrovania spamov môže neurónová sieť zohrať veľkú úlohu, pretože je to systém schopný operatívne prijímať nové vzory, učiť sa, pričom nevyžaduje formalizáciu – pravidlá filtrovania alebo odkláňania dát.

Podobne je zaujímavé predstaviť si, že je daná sieť naučená rozpoznávať dosť veľkú množinu softwarových vírusov. Programové vírusy sú detekčnými algoritmami chápané ako úplne nové aj napriek tomu, že sa veľmi podobajú už známym vírusom. Neurónová sieť je schopná zaradiť program do istej skupiny aj napriek tomu, že sa v ňom nenachádzajú zhody s reprezentantami danej skupiny! Prvok umelej inteligencie takto umožňuje vytvorenie filtračného systému, ktorý isto odchytí známe vírusy, ale zároveň je schopný odchytiť aj vírusy, ktoré sú úplne nové.

5.4 Priateľ počítač

Počítač vybavený prvkami umelej inteligencie, je stroj, ktorý sa v niektorých okamihoch svojej práce musí používateľa pýtať, či je niečo správne alebo nie. Takýto počítač môže čítať obsah obrazovky, sem-tam sa opýta či má niečo zmäkčiť alebo nie, môže byť vybavený lokálnou softvérovou meteorologickou stanicou a predpovedať počasie, hlásiť stav ohrozenia vírusom.

Hendikepovanému človeku môže takto „nahradiť“ priateľa a vytvoriť svet možností, dát, záujmov a novej formy komunikácie človeka a počítača.

[1] Hudec, M.: Informačné technológie v softwarových kompenzačných aplikáciách. Ústav vedy a výskumu UMB, 2006, Banská Bystrica.

[2] Šíma, J. Neruda, R.: Teoretické otázky neuronových sítí. MATFYZPRESS vyd. MfF UK, Praha, 1996.

[3] Nilsson, N. J.: Principles of artificial Intelligence. SPRINGER, 1982.

[4] Hassoun, M. H.: Fundamentals of artificial neural networks. MIT Press, Cambridge, 1995.

[5] SUCHÝ, J., ŠKRINÁROVÁ, J., TRHAN, P.: Neural Network. Modelling of Friction in Robot Joints 8th International Symposium ISMCR 98, CTU and IMEKO. Technical University, Prague, 1998, str. 157-162.

[6] Psutka, J.: Komunikace s počítačem mluvenou řečí. Academia nakladatelství Akademie věd ČR, Praha, 1995.

[7] Jurofský, D.: Speed and Language. Introduction to Natural Language Processing Preatice – Hall, 2000.

[8] T. Dutoid, Y. Stylianou: Handbook of Computational. Linguistics. R. Mitkov, ed., Oxford University Press, 2003

[9] John R. Koza, Martin A. Keane, Matthew J. Streeter, William Mydlowec, Jessen Yu, Guido Lanza: Genetic Programming IV. Routine Human-Competitive Machine Intelligence Kluwer Academic Publishers, 2003.

[10] Siládi, V.: An Optimisation of an Interconnection Network by Parallel Genetic Algorithm on Small Computer Cluster MENDEL 2006. University of Technology Faculty of Automation and Computer Science, Brno 2006, pp. 170-175.

Milan HUDEC
Ústav vedy a výskumu UMB
Banská Bystrica

mail: mhudec@savbb.sk
tel/fax: 048 413 22 36

EFEKTÍVNOSŤ PROGRAMOVACIEHO JAZYKA PRI PRÁCI SO SÚBOROM

V príspevku sa zaoberáme porovnávaním programu v jazyku C a v strojovo orientovanom jazyku (skrátene SOJ). Zameriame sa na operáciu otvorenia a zatvorenia súboru. Porovnáme rýchlosť spracovania týchto operácií, ktoré sú realizované rôznymi spôsobmi v oboch programovacích jazykoch. Výsledky vyhodnotíme.

Úvod

Vo viacerých publikáciách [1, 2, 3, 4] sme sa venovali efektívnosti programu kódovanom vo zvolenom programovacom jazyku. Pre porovnanie sme vybrali jazyk C a jazyk nižšej úrovne – strojovo orientovaný jazyk. Zaujímali sme sa o efektívnosť najčastejšie sa vyskytujúcich operácií v programoch napísaných v uvedených jazykoch. Bolo to kopírovanie, zobrazenie pixelov, výpis čísel a výpis reťazcov. V tomto príspevku sústredíme pozornosť na prácu so súborom .

V jazyku C je viac možností ako otvárať a zatvárať súbory (zatváranie je vždy úzko naviazané na otváranie). Môžeme ho otvoriť buď pomocou štandartnej funkcie *fopen* alebo *open* a zatvoriť pomocou štandartnej funkcie *fclose* alebo *close*.

V strojovo orientovanom jazyku je situácia odlišná. Na prácu so súborom sa poskytuje jeden spôsob otvárania a zatvárania súboru. V nižšie uvedených tabuľkách bude uvedený zistený čas potrebný na otvorenie a následné zatvorenie súboru. Na získanie smerodajných výsledkov sa otváranie a zatváranie súboru vykonávalo 10-krát v jednom behu pre rôzny počet opakovaní (5000, 16000, 23765, 32000). Pri vyberaní počtu opakovaní sme postupovali náhodne. Snahou bolo vybrať ich tak, aby pokryli čo najväčší rozsah a poskytli tak dostatočný rozptyl pre určenie presných časov.

Otvorenie a zatvorenie súboru v SOJ

Uvádzame časť kódu, v ktorom sa 100-krát otvorí a zatvorí súbor, definovaný premennou *sub*.
V registri cx sa nachádza počet opakovaní otvorenia a zatvorenia, do registra alebo je vkladaná značka určujúca spôsob otvárania súboru (v tomto prípade na zápis, ak neexistuje vytvorí sa nový). Premenná kanal predstavuje handler na súbor, cez ktorý sa k súboru pristupuje.

```
mov cx, 100              ;kolkokrat sa subor otvori a zatvori
opakuj:
  mov ah, 3dh
  mov al, 00010010b
  mov dx, seg sub
  mov ds, dx              ;otvorenie suboru
  mov dx, offset sub
  int  21h
  mov kanal, ax
  mov ah, 3eh
  mov bx ,kanal           ;zatvorenie suboru
  int  21h
  loop  opakuj
```

Otvorenie a uzatvorenie súboru v SOJ				
	5000-krát	16000-krát	23765-krát	32000-krát
	0,879120879	2,802197802	4,175824176	5,659340659
	0,879120879	2,802197802	4,175824176	5,659340659
	0,879120879	2,802197802	4,175824176	5,659340659
	0,879120879	2,802197802	4,175824176	5,659340659
	0,879120879	2,802197802	4,175824176	5,659340659
	0,879120879	2,802197802	4,175824176	5,604395604
	0,879120879	2,802197802	4,175824176	5,604395604
	0,879120879	2,802197802	4,175824176	5,604395604
	0,879120879	2,802197802	4,230769231	5,604395604
	0,879120879	2,802197802	4,285714286	5,604395604
priemer	0,879120879	2,802197802	4,192307692	5,631868132
priemer na súbor v ms	0,175824176	0,175137363	0,176406804	0,175995879
priemer za všetky behy	0,175841055			

Priemerná časová hodnota vypovedá o tom, že strojovo orientovaný jazyk je v týchto operáciách naozaj rýchly.

Otvorenie a zatvorenie súboru v jazyku C pomocu fopen a fclose

Príslušný kód v jazyku C je krátky a ľahko pochopiteľný. Podobne ako predchádzajúci kód 100-krát otvára a zatvára súbor s menom “vstup.txt”

```
FILE*fr;
for(i=0; i<100;i++)
  {
   fr = fopen("vstup.txt","r");
   fclose(fr);
  }
```

Otvorenie a uzatvorenie súboru pomocou fopen a fclose v C				
	5000-krát	16000-krát	23765-krát	32000-krát
	2,197802	7,142857	10,549451	14,285714
	2,197802	7,142857	10,549451	14,285714
	2,197802	7,142857	10,659341	14,120879
	2,197802	7,142857	10,659341	14,120879
	2,197802	7,142857	10,659341	14,120879
	2,197802	7,142857	10,659341	14,175824
	2,197802	7,142857	10,604396	14,175824
	2,2252747	7,142857	10,604396	14,175824
	2,2252747	7,032967	10,604396	14,175824
	2,2252747	7,032967	10,604396	14,175824
priemer	2,20604381	7,120879	10,615385	14,1813185
priemer na súbor v ms	0,441208762	0,445054938	0,446681464	0,443166203
priemer za všetky behy	0,444027842			

Všimnime si výsledky. V tomto prípade je jazyk C asi 2,5-krát pomalší, čo nie je pre jazyk C až taký zlý výsledok. V jednotlivých behoch sa vyskytujú minimálne odchýlky .

Otvorenie a zatvorenie súboru v jazyku C pomocou open a close

V tejto ukážke sa 32000-krát otvára a zatvára súbor test.txt.

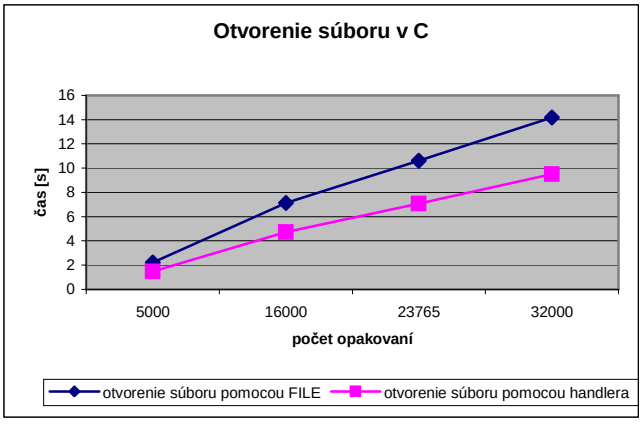
```
int i;
for(i=0; i<32000;i++)
 {

        kanal = open("TEST.TXT", O_CREAT | O_BINARY | O_RDONLY);
         close(kanal);
 }
```

Tento spôsob, podobne ako v strojovo orientovanom programe, získa handler na súbor. Pravdepodobne toto je príčina, prečo je tento spôsob rýchlejší ako predchádzajúci, kde sme získavali ukazovateľ na štruktúru FILE, ktorá už predstavuje nadstavbu nad handler, čo znamená dlhší čas spracovania.

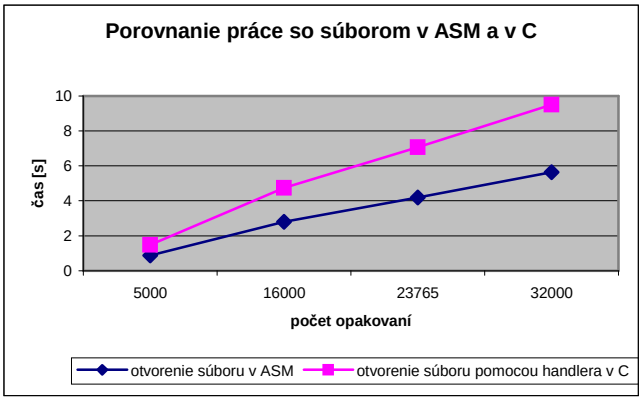
Otvorenie a uzatvorenie súboru pomocou open a close v C				
	5000-krát	**16000-krát**	**23765-krát**	**32000-krát**
	1,483516	4,725275	7,032967	9,505495
	1,483516	4,725275	7,032967	9,505495
	1,483516	4,725275	7,032967	9,505495
	1,483516	4,725275	7,032967	9,505495
	1,483516	4,725275	7,032967	9,505495
	1,483516	4,725275	7,087912	9,505495
	1,483516	4,725275	7,087912	9,505495
	1,483516	4,725275	7,087912	9,505495
	1,483516	4,725275	7,087912	9,505495
	1,428571	4,78022	7,087912	9,505495
priemer	1,4780215	4,7307695	7,0604395	9,505495
priemer na súbor v ms	0,2956043	0,295673094	0,297094025	0,297046719
priemer za všetky behy	0,296354534			

Všimli sme si, že nezáleží na tom, či otvoríme binárny súbor alebo textový súbor. V oboch testoch boli dosiahnuté úplne rovnaké výsledky. Pre porovnanie dĺžky spracovania kódu, v ktorom sa otvára a zatvára súbor uvádzame graf.



Porovnanie rýchlosti práce so súborom v jazyku C a v SOJ

Pre porovnanie sme si zvolili rýchlejší spôsob v jazyku C (pomocou handlera). Na grafe je jasne vidieť rozdiel v rýchlosti práce so súborom v jazyku C a v strojovo orientovanom jazyku (asembler- ASM)



Strojovo orientovaný program je v tomto teste rýchlejší asi 1,5-krát, čo nie je až tak veľa. Dalo by sa očakávať, že bude dosahovať lepšie výsledky. Za povšimnutie stojí najmä takmer totožný lineárny nárast (krivky sú takmer priamky). Pri ideálnom výkone stroja a oboch jazykov by mali byť tieto priamky skutočne lineárne.

Programy v jazyku C boli realizované v prostredí Borlandc. Strojovo orientované programy boli spracované prekladačom TASM a linkerom TLINK. Programy boli spúštané pod tým istým operačným systémom.

Literatúra

1. PARIZKOVA, J.: Porovnávanie operácie kopírovania v dvoch programovacích jazykoch. Informatika v škole, č. 30, str.7-14, www.uips.sk, 2005.
2. PARIZKOVA, J.: Porovnanie rýchlosti zobrazenia bodov obrazovky v dvoch programovacích jazykoch. Informatika v škole, č. 30, str.14-18, www.uips.sk, 2005.
3. PARIZKOVA, J.: Efektívnosť programovacieho jazyka pri výpise čísla. Informatika v škole, č. 30, str.26-33, www.uips.sk, 2005.
4. PARÍZKOVÁ, J.: Efektívnosť programu v jazykoch rôznej úrovne. Technológia vzdelávania, č. 3, str.9-12, 2006.

Jana PARÍZKOVÁ
Fakulta informatiky a informačných technológií STU
Bratislava

KÓDOVANIE VYBRANÝCH OPERÁCIÍ V PROGRAME

V každom programe možno vybrať viac typických operácií, ktoré sa dajú porovnávať podľa istého kritéria z pohľadu použitého programovacieho jazyka.

Vo viacerých publikáciách [1, 2, 3, 4] sme sa venovali efektívnosti programu kódovanom vo zvolenom programovacom jazyku. Pre porovnanie sme vybrali jazyk C a jazyk nižšej úrovne - strojovo orientovaný jazyk (skrátene SOJ). Zaujímali sme sa o efektívnosť niektorých vybraných operácií v programoch napísaných v uvedených jazykoch. Bolo to kopírovanie, zobrazenie pixelov, výpis čísel, výpis reťazcov a práca so súborom.

Tento príspevok sa zaoberá s operáciami volania funkcií, bežnými operáciami so súborom, vstupom a výstupom údajov na obrazovku, ale z iného pohľadu. Nebudeme sa zaujímať o efektívnosť daného kódu, ale o spôsob zápisu príslušnej časti kódu v dvoch programovacích jazykoch a tieto zápisy porovnáme .

Postupne uvedieme kódy uvedených operácií v oboch programovacích jazykoch a budeme hodnotiť iba zložitosť ich zápisu. V strojovo orientovanom jazyku použijeme funkcie na úrovni MS DOS a nie funkcie, ktoré ponúka nižšia vrstva MS BIOS.

1. Volanie funkcií

Pri volaniach funkcií sa využíva na prenos argumentov údajová štruktúra zásobník. Pred zavolaním samotnej funkcie sa vložia potrebné premenné na vrch zásobníka a pri vykonaní funkcie sa zo zásobníka vyberú. Ak funkcia nevracia žiadne argumenty, je treba zásobník vrátiť do pôvodného stavu. V opačnom prípade treba špecifikovať premenné, ktoré bude funkcia vracať.

Zápis funkcie a jej volania v C	**Zápis funkcie a jej volania v SOJ**
Deklarácia: navr. hodnota fun(arg1, arg2,..) { telo } ... ... volanie v programe: fun(arg1, arg2,….);	Deklarácia: fun PROC NEAR PUSH BP MOV BP, SP ;vyberieme argumenty zo zásobníka MOV AX, [BP+4] POP arg1 …. telo procedury … POP BP RET cislo Fun ENDP volanie v programe: PUSH A CALL fun

V SOJ nie je formálne definovaný zápis funkcie, ale procedúry. Inštrukcia RET zabezpečí už spomínaný návrat zásobníka do pôvodného stavu a odovzdanie riadenia do programu, z ktorého bola procedúra volaná inštrukciou CALL. *Cislo* predstavuje počet bajtov, ktoré sme do zásobníka vložili.

2. Práca so súborom

So súborom sa dá podobne ako v C tak aj v SOJ pracovať viacerými spôsobmi. Uvedieme jeden spôsob **otvorenia** existujúceho súboru.

Otvorenie súboru v C	Otvorenie súboru v SOJ
fr = fopen("*subor*","rw");	MOV AH,3dh MOV AL,00010010b MOV DX,seg *data* MOV DS,DX MOV dx,offset *subor* INT 21h MOV *kanal*,ax

V ľavej časti tabuľky „s*ubor*" predstavuje názov súboru, z ktorého chceme čítať aj doňho zapisovať. Premennou „fr" budeme pristupovať k údajom tohoto súboru. V pravej časti tabuľky „ *kanal*" je premenná, pomocou ktorej sa bude pristupovať k dátam v otvorenom súbore. Operačný systém do nej uloží číslo, ktorým sa bude otvorený súbor identifikovať, ak s ním budeme ďalej pracovať. Do registra AL vkladáme číslo spôsobu, akým sa bude k súboru pristupovať (môže byť čítanie, zápis, resp. oboje). V zápise sa predpokladá, že názov súboru ako aj jeho umiestenie v počítači je definovaný premennou „subor" v datovom segmente „data".

Ďalšou operáciou je **čítanie** z otvoreného súboru.

Čítanie súboru v C	Čítanie súboru v SOJ
fscanf(fr,"",…);	MOV AH,3fh MOV BX,*kanal* MOV CX,100 MOV DX,seg *data* MOV DS,DX MOV DX,offset *retazec* INT 21h

Funkciou fscanf zabezpečíme čítanie údajov z predtým otvoreného súboru. V SOJ sa dajú zo súboru čítať iba znaky prípadne reťazce. Programátor si musí upraviť formát pre požadovaný typ údaju. V registri CX je maximálny počet znakov, ktoré sa budú čítať, *retazec* predstavuje miesto v pamäti, kam sa znaky z prečítaného súboru uložia. Samozrejme, že v tomto príklade bude veľkosť premennej *retazec* aspoň 100 znakov. Predpokladáme, že premenná reťazec je definovaná v datovom segmente, ktorý má meno „data".

Na demonštráciu uvedieme ešte **zatvorenie** súboru, s ktorým sa doteraz pracovalo.

V SOJ môžeme uzatvoriť prácu so súborom pomocou funkcie 3eh. Súbor, ktorý uzatvárame je určený číslom v premennej *kanal.*

Zatvorenie súboru v C	Zatvorenie súboru v SOJ
fclose(fr);	MOV AH,3eh MOV BX,*kanal* INT 21h

3. Výstup údajov na obrazovku

V jazyku C ako vyššom programovacom jazyku sú operácie výstupu zapisované pomocou vhodnej funkcie podľa typu vypisovaného údaju. V SOJ existujú štandardné funkcie iba na výpis znaku alebo reťazca znakov. Ak však chceme vypisovať čísla, treba tieto úkony naprogramovať. V nasledujúcich tabuľkách sú zobrazené zodpovedajúce funkcie v C a v SOJ pre výstup údajov rôzneho typu na obrazovku.

Výpis znaku a reťazca

Výpis znaku v C	**Výpis znaku v SOJ**
printf (“%c”,znak) alebo putchar (znak)	MOV DL,znak MOV AH,02h INT 21h
Výpis reťazca v C	**Výpis reťazca v SOJ**
printf (“%s”,retazec); alebo puts (retazec);	MOV DX,offset retazec MOV AH, 09h INT 21h

Samozrejme, že je možné vypísať reťazec opakovaným výpisom znakov v cykle. Tento spôsob je však pomalší (pozri [1]), a preto ho až na špeciálne prípady (chceme nejakým spôsobom reťazec sformátovať) neodporúčame používať.

Výpis čísla

V tejto časti sa nebudeme zaoberať výpisom čísiel s desatinnou časťou.

Pri výpise čísel môžeme naraziť na problém veľkosti čísla. Číslo, ktoré sa zmestí do jednoho 16-bitového registra, čiže jeho veľkosť nepresiahne hodnotu 65535 sa dá jednoducho vypísať v jazyku C funkciou printf. V SOJ treba zabezpečiť výpis reťazca číslic. K tomu môžeme použiť napr. operácie DIV a MOD. Zvyšok po delení daného čísla desiatimi sa ukladá napríklad do zásobníka, a potom sa vypíše ako postupnosť číselných znakov v zodpovedajúcom poradí.

Výpis 16-bitového čísla v C	**Výpis 16-bitového čísla v SOJ**
printf(“%d”,cislo);	OPAK: ;navestie MOV AX,cislo MOV BX,zaklad ;zaklad = 10 XOR DX,DX DIV BX PUSH DX ;vlozime do zasobnika AND AX,AX ; je AX nulove? JNZ OPAK ; ak nie je, opakuj ;tu vyberieme znaky zo zásobníka a vypíšeme

Výpis väčších čísel v SOJ je o niečo náročnejší.

Záver

Z uvedených zápisov zvolených operácií môžeme jednoznačne konštatovať, že kód v jazyku C je omnoho kratší a jednoduchší. Ak však budeme porovnávať efektívnosť daného kódu, potom vo väčšine prípadov bude výsledok v prospech strojovo orientovaného jazyka. Presné výsledky sú v použitej literatúre.

Literatúra

1. PARIZKOVA, J.: Porovnávanie operácie kopírovania v dvoch programovacích jazykoch. Informatika v škole, č. 30, str.7-14, www.uips.sk, 2005.
2. PARIZKOVA, J.: Porovnanie rýchlosti zobrazenia bodov obrazovky v dvoch programovacích jazykoch. Informatika v škole, č. 30, str. 14-18, www.uips.sk, 2005.
3. PARIZKOVA, J.: Efektívnosť programovacieho jazyka pri výpise čísla. Informatika v škole, č. 30, str. 26-33, www.uips.sk, 2005.
4. PARÍZKOVÁ, J.: Efektívnosť programu v jazykoch rôznej úrovne. Technológia vzdelávania, č. 3, str. 9-12, 2006.

Jana PARÍZKOVÁ
Fakulta informatiky a informačných technológií STU
Bratislava

POUŽITIE PROGRAMU EXCEL V TECHNICKEJ VÝUČBE ZAMERANEJ NA MODELOVANIE A ANALÝZU REZNEJ ČASTI VŠEOBECNE ÚČELOVÝCH SKRUTKOVICOVÝCH VRTÁKOV

USING EXCEL SOFTWARE IN TECHNICAL EDUCATION FOCUSED ON MODELLING AND ANALYSIS OF DRILL POINT CUTTING GEOMETRY OF GENETAL PURPOSE TWIST DRILLS

Abstract: *článok popisuje použitie programu Excel v technickej výučbe inžinierskych predmetov zameraných na modelovanie a analýzu reznej časti všeobecne účelových skrutkovicových vrtákov. Program umožňuje študentom robiť rozsiahle počítačové simulácie pomocou menenia rôznych vstupných údajov súvisiacich s geometriou nástroja a reznými podmienkami a sledovať dynamické zmeny v jednotlivých rezných uhloch na nástrojových rezných hranách a ich vplyvu na rezné sily a rezný výkon. Tento prístup odstraňuje potrebu robenia nákladných a časovo náročných experimentálnych testov čím sa znižujú učebné náklady. Zároveň sa zvyšuje efektívnosť výučby, pretože umožňuje študentom pracovať samostatne pri hľadaní optimálnej nástrojovej geometrie pre zvolene pracovné podmienky.*

Kľúčové slová*: Program Excel, technická výučba, počítačové modelovanie, geometria vrtákov.*

Abstract: *the paper describes the ways of using a Type Excel software in technology education based engineering subjects focused on modelling and analysis of drill point geometry of general purpose twist drills. The software allows the students to carry out extensive computer assisted simulations via changes in in-put data associated with the tool geometry and cutting conditions and to observe the dynamic changes occurring in elemental cutting angles along the cutting edges and their effects on the cutting forces and cutting power. This approach eliminates the need of running expensive and time consuming experimental tests, and hence it reduces the investments associated with teaching and learning. In addition, it increases the effectiveness of education because it allows the students to work individually on optimising their tool geometry for chosen cutting conditions.*

Key words*: Excel software, technical education, computer assisted modelling, drill point geometry*

Úvod

V súčasnej dobe počítačové modelovanie je hlavnou zložkou výučby v technicky zameraných odvetviach. Je to preto, že počítač umožňuje rýchle spracovanie zložitých matematických údajov a závislostí. Tento článok popisuje jeden z druhov počítačového použitia pre modelovanie nástrojovej geometrie všeobecne účelového skrutkovicoveho vrtáku a ukazuje príklady rôznych simulácií pre skúmanie „dynamického“ rozloženia jednotlivých uhlov na rezných hranách vrtáku a ich vplyvu na sily a výkon pri rezaní. Praktické príklady sú prezentované v programe Excel, ktorý je bežne dostupný na stredných aj vysokých školách. Zložitejší variant tohoto programovania vytvorený v programovacom jazyky Fortran je popísaný vzorcami a algoritmom v článku [1], a overený v praxi [2 – 4] pri robení výskumu na báze doktoranského štúdia [5, 6].

1. Metodika a rozbor problému z matematického pohľadu

Na stredných a vysokých školách praktická výučba zameraná na výrobu preberá podrobne geometriu jednotlivých nástrojov a ich správanie sa počas rezania. Je to preto, že v súčasnosti strojové centrá vo výrobe a v laboratóriách sú väčšinou vybavené CNC strojmi a automatmi, ktoré používajú rôzne rezné nástroje. Vŕtanie je obvykle poslednou výrobnou operáciou pri dokončovaní výrobkov, preto sa dôraz kladie na vrtáky, aby sa nezlomil a neznehodnotil výrobok a čas a náklady vložené na jeho výrobu. Strojové centra sú stavané na masovú výrobu. Vrtáky, ako aj iné rezné nástroje, musia pracovať na vysokých rezných rýchlostiach posuvoch, odolávať silám a vibráciám. Modelovanie popísané v tejto kapitole poukazuje na to, ako využiť počítač pre optimalizáciu a výučbu, prípadne výskum súvisiaci s vrtákmi. Dole popísaný model je vhodný pre aplikovanie do výučby na stredných aj vysokých školách.

Tabuľka 1 ukazuje vstupný druh údajov pre počítačové modelovanie. Údaje sú v angličtine z toho dôvodu, ak by čitateľ potreboval použiť referenčné materiály z časopisov [1-3] ktoré sú v angličtine. Význam symbolov je preložený do slovenčiny vedľa tabuľky.

Tabuľka 1: Vstupné údaje použité pre počítačové modelovanie

A	B	C	D	E	F	G
2						
3	SOFTWARE FOR CALCULATING THE RAKE ANGLES ALONG THE LIP AND CHISEL EDGE					
4						
5	I.	DRILL TYPE:	GENERAL PURPOSE TWIST DRILL			
6		IN-PUT DATA			Real Values	
7		DRILL FEATURES	SYMBOL	UNIT	in put	softw. code
8	1	DIAMETER	ϕD	[mm]	6.35	D
9	2	LIP SPACING	2W	[mm]	0.965	W2
10	3	POINT ANGLE	2P	[deg]	115	P2
11	4	CHISEL EDGE ANGLE	ψ	[deg]	123	CHI
12	5	LIP CLEARANCE	Clo	[deg]	12.3	CLO
13	6	HELIX ANGLE	δo	[deg]	26.3	DELTAD
14	7	SETTING ANGLE	ϕ	[deg]	30	FI
15						
16		CUTTING PARAMETERS				
17	1	peripheral speed	v	[m/min]	15.072	PSPEED
18	2	feed	f	[mm/rev]	0.149	FEED
19						
20		NUMBER OF ELEMENTS				
21	1	LIP	25			M
22	2	CHISEL	25			MC
23						
24		GRINDING DETAILS				
25	1	NONTHINNED	THIN=0		0	
26	2	THINNED	THIN=1		1	

Symboly:

D - priemer vrtaku *i.e.* drill diameter

ψ - uhol priecneho ostria *i.e.* chisel edge angle

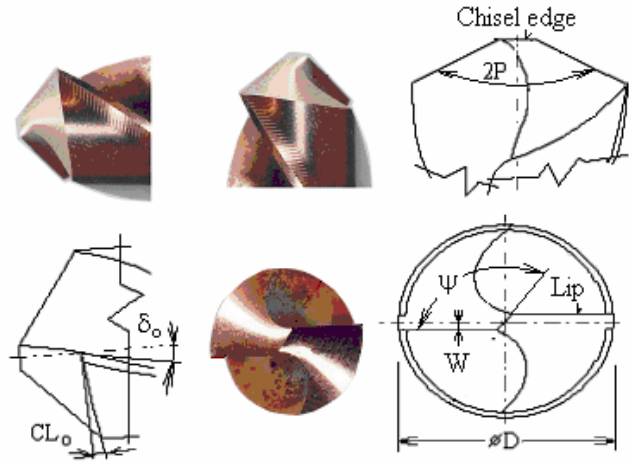
2P - uhol klinu *i.e.* point angle

δ_o - uhol sklonu draziek *i.e.* helix angle

Cl_o - uhol chrbta *i.e.* body clearance

2W - telesova hrubka vrtaku *i.e.* web thickness

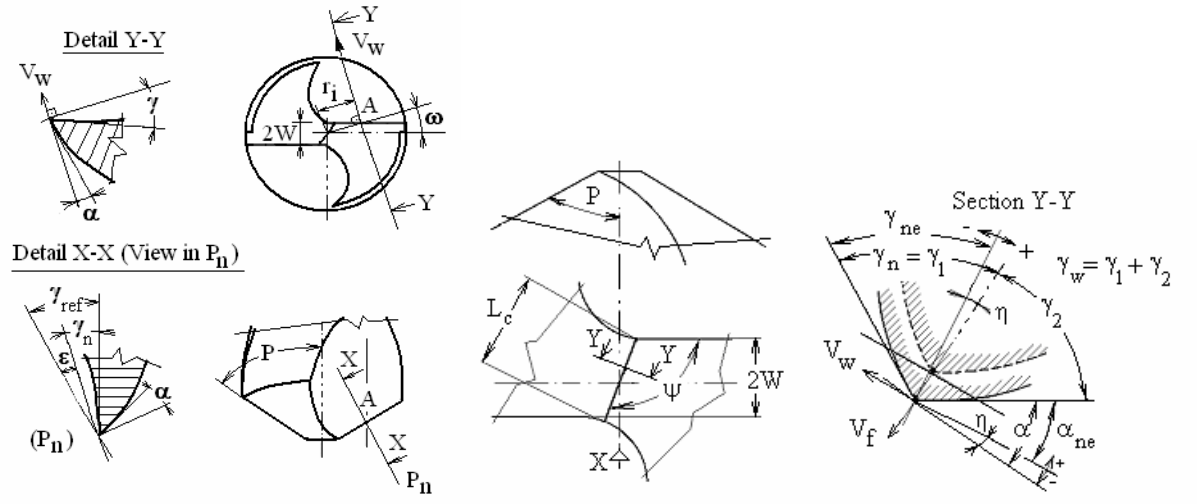
Obrázok 1 ukazuje základné geometrické veličiny: D, ψ, 2P, δ_o, Cl_o, 2W, tvoriace hlavnú hrotovú geometriu skrutkovicových vrtákov podieľajúcu sa na rezaní. Tabuľka vpravo pri obrázku 1 ukazuje doplnkové uhly a vzťahy medzi hlavnými geometrickými veličinami vrtákovej statickej geometrie.



H	I	J	K	L
	SOFTWARE VAR.	softw. code	Values	Excel Equations
8	PI		3.141593	= 4*ATAN(1)
9	RAD		0.017453	= K8/180
10	DRILL RADIUS	R	3.175	= F8/2
11	1/2 LIP SPACING	W	0.4825	= F9/2
12	2W/ϕD ratio	WOR	0.151969	= F9/F8
13				
14	chisel doplnkovy	CHID	57	= 180-F11
15	chisel doplnkovy (rad)	CHIDR	0.994838	= K14*K9
16		WC	0.313339	= K11/(TAN(K15))
17	half of point angle	P	57.5	= F10/2
18	half of point angle (rad)	PR	1.003564	= K17*K9
19	helix angle (rad)	DELTDR	0.459022	= F13*K9
20	lip clearance (rad)	CLOR	0.214675	= F12*K9
21	setting angle (rad)	FIR	0.523599	= F14*K9
22	LEAD	LEAD	40.36396	=(K8*F8)/(TAN(K19))
23	revolutions	SPEED	755.5223	= (1000*F17)/(K8*F8)
24	TOP		0.259247	= K11*COS(K18)

Obrázok 1: Geometria všeobecne účelového skrutkovicového vrtáku a s ňou súvisiace vzťahy

Pri rezaní sa statická geometria vrtáka začne správať ako dynamická. Rezné hrany (lip) a (chisel edge) v hrotovej geometrii vrtáku v jednotlivých elementárnych úsekoch nadobúdajú rôznu geometriu a ich statické uhly sa menia na dynamické ako je to naznačené v obrázku 2.



Obrázok 2: Nákres ukazujúci zmenu statickej geometrie na pracovnú pre rezné hrany: (a) v 'lip' regióne, a (b) v 'chisel' regióne, skrutkovicového vrtáku [1, 5].

Matematický zápis geometrických závislostí a vzájomných vzťahov medzi statickými a dynamickými veličinami a ich vplyvom na rezné sily a rezný výkon je možné zapísať nasledovným spôsobom, ktorý bol spracovaný významným odborníkom v oblasti obrábania Armaregom [7], ktorý bol vedúcim úlohy [5] riešenej autorom tohoto článku v rámci jeho výskumného študijného pobytu na Melbornskej Univerzite:

Rezná sila v elemntárnych bodoch na reznej hrane (v lip regióne)

$$T_{hl} = \sum_{j=1}^{M} \delta T_{hlj} \qquad (1)$$

Kde:

$$\delta T_{hlj} = \left[\left(\delta F_Q + \delta F_{Qe} \right).\cos\varepsilon.\sin p - \left(\delta F_R + \delta F_{\mathrm{Re}} \right).\left(\cos\lambda_s.\cos p + \sin\lambda_s.\sin p.\sin\varepsilon \right) \right]$$

$$\tan\varepsilon = \frac{V_W.\sin\omega.\cos p}{V_W.\cos\omega} = \tan\omega.\cos p$$

$\omega = \sin^{-1}(W/r_i)$

<u>*Momentová rezná sila a jej elementárne zložky*</u>:

$$T_{ql} = 2\sum_{j=1}^{M} \delta T_{qlj} \quad \text{kde} \quad \delta T_{qlj} = r_i.\left(\delta F_P + \delta F_{Pe} \right) \qquad (2)$$

Dĺžka reznej hrany a jej elementárne zložky:

$$L = 2\sum_{j=1}^{M} \delta L \qquad \text{keď} \qquad \delta L = \frac{\frac{\phi D}{2}.\cos\omega_o - W.\cot(180°-\psi)}{M.\sin p} \tag{3}$$

uhlová rýchlosť (ω_o) je daná, (ψ) je meraná hodnota.

Uvažovaný bod na reznej hrane $r_i = \sqrt{\left[\frac{\phi D}{2}.\cos\omega_o - (i-\frac{1}{2}).\delta L.\sin p\right]^2 + W^2}$ (4)

Hrúbka rezu: $t = \frac{f.\sin p.\cos\varepsilon}{2}$ (5)

Jednotlivá šírka rezu

$\delta b = \delta L.\cos\lambda_s$ kde $\lambda_s = \frac{V_W.\sin\omega.\sin p}{V_W} = \sin\omega.\sin p$ (6)

Jednotlivá plocha rezu: $\delta A = t.\delta b$ (7)

Jednotlivé rezné zložky:

$$\delta F_P = \frac{\tau.\delta A.\left[\cos(\beta_n-\gamma_n).\cos\lambda_s + \tan\eta_c.\sin\lambda_s.\sin\beta_n\right]}{\sqrt{\cos^2.(\phi_n+\beta_n-\gamma_n)+\tan^2\eta_c.\sin^2\beta_n}.(\sin\phi_n.\cos\lambda_s)} \tag{8}$$

$$\delta F_Q = \frac{\tau.\delta A.\sin.(\beta_n-\gamma_n)}{\sqrt{\cos^2.(\phi_n+\beta_n-\gamma_n)+\tan^2\eta_c.\sin^2\beta_n}.(\sin\phi_n.\cos\lambda_s)} \tag{9}$$

$$\delta F_R = \frac{\tau.\delta A.\left[\cos(\beta_n-\gamma_n).\sin\lambda_s - \tan\eta_c.\cos\lambda_s.\cos\beta_n\right]}{\sqrt{\cos^2.(\phi_n+\beta_n-\gamma_n)+\tan^2\eta_c.\sin^2\beta_n}.(\sin\phi_n.\cos\lambda_s)} \tag{10}$$

kde $\tan\beta_n = \tan\beta.\cos\eta_c$ a (γ_n) moze byt pocitana zo vztahov 11 do 18.

$$\tan\gamma_n = \frac{\tan\delta.\left[\left(\cos\omega+\sin\omega.\tan\omega.\cos^2 p\right)-\tan\omega.\cos p\right]}{\sin p} \tag{11}$$

$$\tan\gamma_n = \frac{\sin\omega_o.\tan.\delta_o.\left[\left(\cot\omega+\tan\omega.\cos^2 p\right)-\tan\omega.\cos p\right]}{\sin p} \tag{12}$$

$$\tan\gamma_n = \frac{\tan\delta_o.(r^2 - W^2.\sin^2 p) - \frac{\phi D}{2}.W.\sin p.\cos p}{\frac{\phi D}{2}.\sin.\sqrt{r^2 - W^2}} \quad (13)$$

$$\tan\phi_n = \frac{r_l.(\cos\eta_c / \cos\lambda_s).\cos\gamma_n}{1 - r_l.(\cos\eta_c.\cos\lambda_s).\sin\gamma_n} \quad (14)$$

$$\tan(\phi_n + \beta_n) = \frac{\tan\lambda_s.\cos\gamma_n}{\tan\eta_c - \sin\gamma_n.\tan\lambda_s} \quad (15)$$

$$\delta F_{Pe} = K_{1P}.\delta b \quad (16)$$

$$\delta F_{Qe} = K_{1Q}.\delta b \quad (17)$$

$$\delta F_{\mathrm{Re}} = K_{1R}.\delta b = K_{1P}.\sin\lambda_s.\delta b \qquad \text{alebo } \delta F_{re} \cong 0 \quad (18)$$

Tlačná sila vo fazetke (chisel regióne)

$$T_{hc} = \sum_{k=1}^{M} \delta T_{hck} \qquad \text{where} \qquad \delta T_{hck} = \delta F_{Pc}.\sin\eta + \delta F_{Qc}.\cos\eta \quad (19)$$

Krútiaca sila a jej jednotlivé zložky

$$T_{qc} = 2\sum_{k=1}^{M} \delta T_{qck} \quad (20)$$

kde $\delta T_{qck} = r_k.(\delta F_{Pc}.\cos\eta - \delta F_{Qc}.\sin\eta)$ a $r_k = r_c - (k - \frac{1}{2}).\delta b = \frac{L_c}{2} - (k - \frac{1}{2}).\delta b$

a $\delta b = \frac{r_c - r_{\lim it.}}{M} = \frac{L_c - 2.r_{\lim it.}}{2.M_c}$

Kde $r_{limit.}$ je polomer pre α_{ne} (=0), η (=90°-γ_2), posuv (f), and uhol $\gamma_W = \gamma_1 + \gamma_2$.

Vtláčacia sila $T_{hI} = \delta T_{hck}.\left(\frac{2.r_{\lim it.}}{\delta b}\right)$ (21)

Hrúbka rezu $t = \frac{f.\cos\eta}{2}$ (22)

Dĺžka hrany z nástrojovej geometrie: $L_c = \frac{2W}{\sin(180° - \psi)}$ (23)

Jednotlivé zložky síl

$$\delta F_{Pc} = C_{IP}.\delta b \qquad (23)$$

$$\delta F_{Qc} = C_{IQ}.\delta b \qquad (24)$$

Ako vidieť z horeuvedenej geometrickej a matematickej analýzy vzťahy sú dosť zložité. Študenti so slabšími vedomosťami v matematike a geometrii by zjavne mali problémy s pochopením takých závislostí bez možností použitia počítača.

2. Praktické ukážky použitia Excelu vo výučbe zameranej na optimalizáciu a modelovanie rezných vrtacích nástrojov

Ďalšie ukážky sú z použitia programu Excel pre zápis vzťahov 1 – 24 a ich praktického použitia pre modelovanie v počítačom podporovanej výučbe v oblasti strojársensky zameraných predmetov na výrobu a optimalizáciu.

Tabuľka 2 ukazuje praktické – vypočítané – údaje (vľavo) zo vzťahov zapísaných v kódovom systéme pre Excel (vpravo) pre hlavný (lip) rezný región. Tabuľka 3 ukazuje opäť praktické – vypočíané – údaje (vľavo) zo vzťahov zapísaných v kódovom systéme pre Excel (vpravo) pre fazetku (chisel rezný región).

Modelovanie je robené pomocou menenia vstupných údajov v tabuľke 1, čiže zmenami v bodovej vrtákovej geometrii: D (F8); 2W (F9); 2P (F10); ψ (F11); Cl_o, (F12); δ_o (F13); a v rezných parametroch: v (F17); f (F18). Pre tieto vstupné údaje počítač prepočíta elementálnu distribúciu pracovných uhlov na reznom kline pre rezné hrany, a zobrazí to graficky ako je to ukázané na obrázku 3.

Tabuľka 2: Praktické ukážky použitia Excelu pre výpočet elementálnych pracovných uhlov (a ich distribúcie) pozdĺž hlavnej reznej hrany (pre lip región)

M	N	O	P	Q	R	S	T	U	V	W
2										
3										
4	M=	25		DL	0.112991	equals to	(SQRT(K10^2-K11^2)-K16)/O4			
5										
6	SOFTWARE (LIP)									
7	AN=AM	from 1 to 25								
8	AN(1)	25	DL(1)	0.112991	RNA(1)	0.369835	RN(1)	0.607934	v(1)	2.885917
9	AN(2)	24	DL(2)	0.112991	RNA(2)	0.482826	RN(2)	0.682589	v(2)	3.240308
10	AN(3)	23	DL(3)	0.112991	RNA(3)	0.595818	RN(3)	0.915149	v(3)	4.344294
11	AN(4)	22	DL(4)	0.112991	RNA(4)	0.708809	RN(4)	0.992426	v(4)	4.711134
12	AN(5)	21	DL(5)	0.112991	RNA(5)	0.8218	RN(5)	1.076037	v(5)	5.108041
13	AN(6)	20	DL(6)	0.112991	RNA(6)	0.934792	RN(6)	1.164618	v(6)	5.528543
14	AN(7)	19	DL(7)	0.112991	RNA(7)	1.047783	RN(7)	1.257119	v(7)	5.967655
15	AN(8)	18	DL(8)	0.112991	RNA(8)	1.160774	RN(8)	1.352737	v(8)	6.42156
16	AN(9)	17	DL(9)	0.112991	RNA(9)	1.273766	RN(9)	1.450855	v(9)	6.887333
17	AN(10)	16	DL(10)	0.112991	RNA(10)	1.386757	RN(10)	1.550998	v(10)	7.362723
18	AN(11)	15	DL(11)	0.112991	RNA(11)	1.499749	RN(11)	1.652799	v(11)	7.845982
19	AN(12)	14	DL(12)	0.112991	RNA(12)	1.61274	RN(12)	1.75597	v(12)	8.335741
20	AN(13)	13	DL(13)	0.112991	RNA(13)	1.725731	RN(13)	1.860282	v(13)	8.830919
21	AN(14)	12	DL(14)	0.112991	RNA(14)	1.838723	RN(14)	1.965554	v(14)	9.330653
22	AN(15)	11	DL(15)	0.112991	RNA(15)	1.951714	RN(15)	2.071639	v(15)	9.834249
23	AN(16)	10	DL(16)	0.112991	RNA(16)	2.064705	RN(16)	2.178419	v(16)	10.34114
24	AN(17)	9	DL(17)	0.112991	RNA(17)	2.177697	RN(17)	2.285796	v(17)	10.85087
25	AN(18)	8	DL(18)	0.112991	RNA(18)	2.290688	RN(18)	2.39369	v(18)	11.36305
26	AN(19)	7	DL(19)	0.112991	RNA(19)	2.40368	RN(19)	2.502034	v(19)	11.87737
27	AN(20)	6	DL(20)	0.112991	RNA(20)	2.516671	RN(20)	2.610772	v(20)	12.39356
28	AN(21)	5	DL(21)	0.112991	RNA(21)	2.629662	RN(21)	2.719857	v(21)	12.9114
29	AN(22)	4	DL(22)	0.112991	RNA(22)	2.742654	RN(22)	2.829249	v(22)	13.43069
30	AN(23)	3	DL(23)	0.112991	RNA(23)	2.855645	RN(23)	2.938913	v(23)	13.95127
31	AN(24)	2	DL(24)	0.112991	RNA(24)	2.968636	RN(24)	3.04882	v(24)	14.47301
32	AN(25)	1	DL(25)	0.112991	RNA(25)	3.081628	RN(25)	3.158944	v(25)	14.99578

Kódový systém od Q8 do Q32 je rovny R4 čiže Q8=R4; Q9=R4; … Q32=R4

Kde hodnota R4 je elemntálna dĺžka (1/25 tina celkovej dĺžky) reznej hrany.

Kódový systém S8 do S32 je počítaný nasledovne

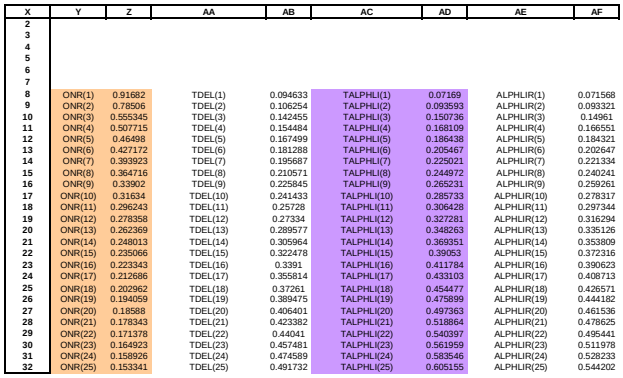
pre S8=SQRT(K10^2-K11^2)-(O8-0.5)*Q8; …..; a pre S32=SQRT(K10^2-K11^2)-(O32-0.5)*Q32

Kódový systém U8 do U32 je počítaný nasledovne:

pre U8=SQRT(S8^2+K11^2); …..; a pre U32=SQRT(S32^2+K11^2)

Kódový systém pre W8 do W32 je počítaný nasledovne:

pre W8=2*K8*U8*K23*0.001; …; a pre W32= 2*K8*U32*K23*0.001



X	Y	Z	AA	AB	AC	AD	AE	AF
2								
3								
4								
5								
6								
7								
8	ONR(1)	0.91682	TDEL(1)	0.094633	TALPHLI(1)	0.07169	ALPHLIR(1)	0.071568
9	ONR(2)	0.78506	TDEL(2)	0.106254	TALPHLI(2)	0.093593	ALPHLIR(2)	0.093321
10	ONR(3)	0.555345	TDEL(3)	0.142455	TALPHLI(3)	0.150736	ALPHLIR(3)	0.14961
11	ONR(4)	0.507715	TDEL(4)	0.154484	TALPHLI(4)	0.168109	ALPHLIR(4)	0.166551
12	ONR(5)	0.46498	TDEL(5)	0.167499	TALPHLI(5)	0.186438	ALPHLIR(5)	0.184321
13	ONR(6)	0.427172	TDEL(6)	0.181288	TALPHLI(6)	0.205467	ALPHLIR(6)	0.202647
14	ONR(7)	0.393923	TDEL(7)	0.195687	TALPHLI(7)	0.225021	ALPHLIR(7)	0.221334
15	ONR(8)	0.364716	TDEL(8)	0.210571	TALPHLI(8)	0.244972	ALPHLIR(8)	0.240241
16	ONR(9)	0.33902	TDEL(9)	0.225845	TALPHLI(9)	0.265231	ALPHLIR(9)	0.259261
17	ONR(10)	0.31634	TDEL(10)	0.241433	TALPHLI(10)	0.285733	ALPHLIR(10)	0.278317
18	ONR(11)	0.296243	TDEL(11)	0.25728	TALPHLI(11)	0.306428	ALPHLIR(11)	0.297344
19	ONR(12)	0.278358	TDEL(12)	0.27334	TALPHLI(12)	0.327281	ALPHLIR(12)	0.316294
20	ONR(13)	0.262369	TDEL(13)	0.289577	TALPHLI(13)	0.348263	ALPHLIR(13)	0.335126
21	ONR(14)	0.248013	TDEL(14)	0.305964	TALPHLI(14)	0.369351	ALPHLIR(14)	0.353809
22	ONR(15)	0.235066	TDEL(15)	0.322478	TALPHLI(15)	0.39053	ALPHLIR(15)	0.372316
23	ONR(16)	0.223343	TDEL(16)	0.3391	TALPHLI(16)	0.411784	ALPHLIR(16)	0.390623
24	ONR(17)	0.212686	TDEL(17)	0.355814	TALPHLI(17)	0.433103	ALPHLIR(17)	0.408713
25	ONR(18)	0.202962	TDEL(18)	0.37261	TALPHLI(18)	0.454477	ALPHLIR(18)	0.426571
26	ONR(19)	0.194059	TDEL(19)	0.389475	TALPHLI(19)	0.475899	ALPHLIR(19)	0.444182
27	ONR(20)	0.18588	TDEL(20)	0.406401	TALPHLI(20)	0.497363	ALPHLIR(20)	0.461536
28	ONR(21)	0.178343	TDEL(21)	0.423382	TALPHLI(21)	0.518864	ALPHLIR(21)	0.478625
29	ONR(22)	0.171378	TDEL(22)	0.44041	TALPHLI(22)	0.540397	ALPHLIR(22)	0.495441
30	ONR(23)	0.164923	TDEL(23)	0.457481	TALPHLI(23)	0.561959	ALPHLIR(23)	0.511978
31	ONR(24)	0.158926	TDEL(24)	0.474589	TALPHLI(24)	0.583546	ALPHLIR(24)	0.528233
32	ONR(25)	0.153341	TDEL(25)	0.491732	TALPHLI(25)	0.605155	ALPHLIR(25)	0.544202

Kódový systém pre Z8 do Z32 je počítaný nasledovne:

pre Z8=ASIN(K11/U8); …; a pre Z32= ASIN(K11/U32)

Kódový systém pre AB8 do AB32 je počítaný nasledovne:

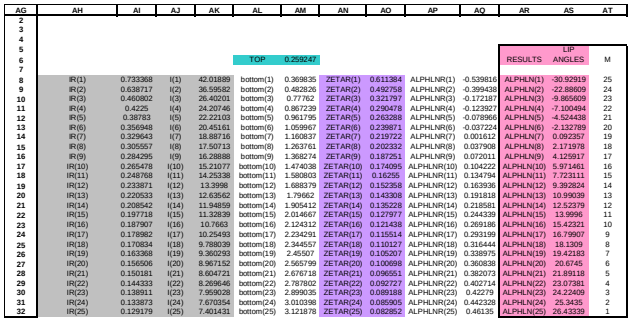
pre AB8=2*K8*U8/K22; ; a pre AB32==2*K8*U32/K22

pre AD8=AB8*COS(Z8)/(SIN(K18)-AB8*SIN(Z8)*COS(K18))

a pre

AD32=AB32*COS(Z32)/(SIN(K18)-AB32*SIN(Z32)*COS(K18))

pre AF8=ATAN(AD8);…;pre AF32=ATAN(AD32)



AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT
2													
3													
4													
5												LIP	
6					TOP	0.259247					RESULTS	ANGLES	M
7													
8	IR(1)	0.733368	I(1)	42.01889	bottom(1)	0.369835	ZETAR(1)	0.611384	ALPHLNR(1)	-0.539816	ALPHLN(1)	-30.92919	25
9	IR(2)	0.638717	I(2)	36.59582	bottom(2)	0.482826	ZETAR(2)	0.492758	ALPHLNR(2)	-0.39943	ALPHLN(2)	-22.88609	24
10	IR(3)	0.460802	I(3)	26.40201	bottom(3)	0.77762	ZETAR(3)	0.321797	ALPHLNR(3)	-0.172187	ALPHLN(3)	-9.865609	23
11	IR(4)	0.4225	I(4)	24.20746	bottom(4)	0.867239	ZETAR(4)	0.290478	ALPHLNR(4)	-0.123927	ALPHLN(4)	-7.100494	22
12	IR(5)	0.38783	I(5)	22.22103	bottom(5)	0.961795	ZETAR(5)	0.263288	ALPHLNR(5)	-0.07966	ALPHLN(5)	-4.524438	21
13	IR(6)	0.356948	I(6)	20.45161	bottom(6)	1.059967	ZETAR(6)	0.239871	ALPHLNR(6)	-0.037224	ALPHLN(6)	-2.132789	20
14	IR(7)	0.329643	I(7)	18.88716	bottom(7)	1.160837	ZETAR(7)	0.219722	ALPHLNR(7)	0.001612	ALPHLN(7)	0.092357	19
15	IR(8)	0.305557	I(8)	17.50713	bottom(8)	1.263761	ZETAR(8)	0.202332	ALPHLNR(8)	0.037908	ALPHLN(8)	2.171978	18
16	IR(9)	0.284295	I(9)	16.28888	bottom(9)	1.368274	ZETAR(9)	0.187251	ALPHLNR(9)	0.072011	ALPHLN(9)	4.125917	17
17	IR(10)	0.265478	I(10)	15.21077	bottom(10)	1.474038	ZETAR(10)	0.174095	ALPHLNR(10)	0.104222	ALPHLN(10)	5.971461	16
18	IR(11)	0.248768	I(11)	14.25338	bottom(11)	1.580803	ZETAR(11)	0.16255	ALPHLNR(11)	0.134794	ALPHLN(11)	7.723111	15
19	IR(12)	0.233871	I(12)	13.3998	bottom(12)	1.688379	ZETAR(12)	0.152358	ALPHLNR(12)	0.163936	ALPHLN(12)	9.392824	14
20	IR(13)	0.220533	I(13)	12.63562	bottom(13)	1.79662	ZETAR(13)	0.143308	ALPHLNR(13)	0.191818	ALPHLN(13)	10.99039	13
21	IR(14)	0.208542	I(14)	11.94859	bottom(14)	1.905412	ZETAR(14)	0.135228	ALPHLNR(14)	0.218581	ALPHLN(14)	12.52379	12
22	IR(15)	0.197718	I(15)	11.32839	bottom(15)	2.014667	ZETAR(15)	0.127977	ALPHLNR(15)	0.244339	ALPHLN(15)	13.9996	11
23	IR(16)	0.187907	I(16)	10.7663	bottom(16)	2.124312	ZETAR(16)	0.121438	ALPHLNR(16)	0.269186	ALPHLN(16)	15.42321	10
24	IR(17)	0.178982	I(17)	10.25493	bottom(17)	2.234291	ZETAR(17)	0.115514	ALPHLNR(17)	0.293199	ALPHLN(17)	16.79907	9
25	IR(18)	0.170834	I(18)	9.788039	bottom(18)	2.344557	ZETAR(18)	0.110127	ALPHLNR(18)	0.316444	ALPHLN(18)	18.1309	8
26	IR(19)	0.163368	I(19)	9.360293	bottom(19)	2.45507	ZETAR(19)	0.105207	ALPHLNR(19)	0.338975	ALPHLN(19)	19.42183	7
27	IR(20)	0.156506	I(20)	8.967152	bottom(20)	2.565799	ZETAR(20)	0.100698	ALPHLNR(20)	0.360838	ALPHLN(20)	20.6745	6
28	IR(21)	0.150181	I(21)	8.604721	bottom(21)	2.676718	ZETAR(21)	0.096551	ALPHLNR(21)	0.382073	ALPHLN(21)	21.89118	5
29	IR(22)	0.144333	I(22)	8.269646	bottom(22)	2.787802	ZETAR(22)	0.092727	ALPHLNR(22)	0.402714	ALPHLN(22)	23.07381	4
30	IR(23)	0.138911	I(23)	7.959028	bottom(23)	2.899035	ZETAR(23)	0.089188	ALPHLNR(23)	0.42279	ALPHLN(23)	24.22409	3
31	IR(24)	0.133873	I(24)	7.670354	bottom(24)	3.010398	ZETAR(24)	0.085905	ALPHLNR(24)	0.442328	ALPHLN(24)	25.3435	2
32	IR(25)	0.129179	I(25)	7.401431	bottom(25)	3.121878	ZETAR(25)	0.082852	ALPHLNR(25)	0.46135	ALPHLN(25)	26.43239	1

pre AI8=ASIN(SIN(Z8)*SIN(K18));…; a pre AI32=ASIN(SIN(Z32)*SIN(K18))

pre AK8=AI8/A9; a pre AK32=AI32/A9

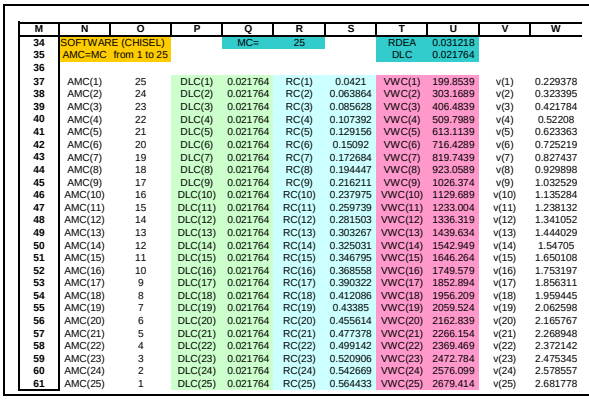
pre AM8=SQRT(U8^2-K11^2;…; a pre AM32=SQRT(U32^2-K11^2

pre AO8=ATAN(AM6/AM8); a pre AO32=ATAN(AM6/AM32)

pre AQ8=AF8-AO8; …; a pre AQ32=AF32-AO32

pre AS8=AQ8/K9;…; a pre AS32=AQ32/K9

Tabuľka 3: Praktické ukážky použitia Excelu pre výpočet elementálnych pracovných uhlov (a ich distribúcie) pozdĺž vedľajšej reznej hrany (pre chisel región)



M	N	O	P	Q	R	S	T	U	V	W
34	SOFTWARE (CHISEL)			MC=	25		RDEA	0.031218		
35	AMC=MC from 1 to 25						DLC	0.021764		
36										
37	AMC(1)	25	DLC(1)	0.021764	RC(1)	0.0421	VWC(1)	199.8539	v(1)	0.229378
38	AMC(2)	24	DLC(2)	0.021764	RC(2)	0.063864	VWC(2)	303.1689	v(2)	0.323395
39	AMC(3)	23	DLC(3)	0.021764	RC(3)	0.085628	VWC(3)	406.4839	v(3)	0.421784
40	AMC(4)	22	DLC(4)	0.021764	RC(4)	0.107392	VWC(4)	509.7989	v(4)	0.52208
41	AMC(5)	21	DLC(5)	0.021764	RC(5)	0.129156	VWC(5)	613.1139	v(5)	0.623363
42	AMC(6)	20	DLC(6)	0.021764	RC(6)	0.15092	VWC(6)	716.4289	v(6)	0.725219
43	AMC(7)	19	DLC(7)	0.021764	RC(7)	0.172684	VWC(7)	819.7439	v(7)	0.827437
44	AMC(8)	18	DLC(8)	0.021764	RC(8)	0.194447	VWC(8)	923.0589	v(8)	0.929898
45	AMC(9)	17	DLC(9)	0.021764	RC(9)	0.216211	VWC(9)	1026.374	v(9)	1.032529
46	AMC(10)	16	DLC(10)	0.021764	RC(10)	0.237975	VWC(10)	1129.689	v(10)	1.135284
47	AMC(11)	15	DLC(11)	0.021764	RC(11)	0.259739	VWC(11)	1233.004	v(11)	1.238132
48	AMC(12)	14	DLC(12)	0.021764	RC(12)	0.281503	VWC(12)	1336.319	v(12)	1.341052
49	AMC(13)	13	DLC(13)	0.021764	RC(13)	0.303267	VWC(13)	1439.634	v(13)	1.444029
50	AMC(14)	12	DLC(14)	0.021764	RC(14)	0.325031	VWC(14)	1542.949	v(14)	1.54705
51	AMC(15)	11	DLC(15)	0.021764	RC(15)	0.346795	VWC(15)	1646.264	v(15)	1.650108
52	AMC(16)	10	DLC(16)	0.021764	RC(16)	0.368558	VWC(16)	1749.579	v(16)	1.753197
53	AMC(17)	9	DLC(17)	0.021764	RC(17)	0.390322	VWC(17)	1852.894	v(17)	1.856311
54	AMC(18)	8	DLC(18)	0.021764	RC(18)	0.412086	VWC(18)	1956.209	v(18)	1.959445
55	AMC(19)	7	DLC(19)	0.021764	RC(19)	0.43385	VWC(19)	2059.524	v(19)	2.062598
56	AMC(20)	6	DLC(20)	0.021764	RC(20)	0.455614	VWC(20)	2162.839	v(20)	2.165767
57	AMC(21)	5	DLC(21)	0.021764	RC(21)	0.477378	VWC(21)	2266.154	v(21)	2.268948
58	AMC(22)	4	DLC(22)	0.021764	RC(22)	0.499142	VWC(22)	2369.469	v(22)	2.372142
59	AMC(23)	3	DLC(23)	0.021764	RC(23)	0.520906	VWC(23)	2472.784	v(23)	2.475345
60	AMC(24)	2	DLC(24)	0.021764	RC(24)	0.542669	VWC(24)	2576.099	v(24)	2.578557
61	AMC(25)	1	DLC(25)	0.021764	RC(25)	0.564433	VWC(25)	2679.414	v(25)	2.681778

Q37=U35;…Q61=U35

S37=K11/SIN(K15)-(O37-0.5)*Q37;…; S61= K11/SIN(K15)-(O61-0.5)*Q61

U37= 2*K8*S37*K23;…; U61= 2*K8*S61*K23

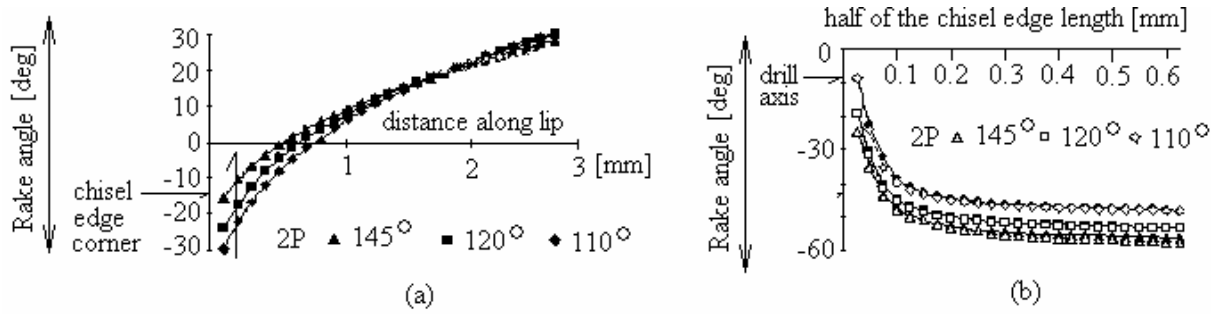
W37= SQRT(U37^2+Y34^2))*0.001;…;W61= (SQRT(U61^2+Y34^2))*0.001

X	Y	Z	AA	AB	AC	AD
VFC=	112.572823		ALPHNR	-0.921167		
TALPHNR	1.31644909		ALPHN	-52.779		
					RESULTS: CHISEL EDGE ANGLES	
37	ZETAR(1)	0.512978	ALPHLNR (1) unthinned	-0.408189	ALPHN (1) unthinned	-23.3875
38	ZETAR(2)	0.355541	ALPHLNR (2) unthinned	-0.565626	ALPHN (2) unthinned	-32.40801
39	ZETAR(3)	0.270172	ALPHLNR (3) unthinned	-0.650996	ALPHN (3) unthinned	-37.29931
40	ZETAR(4)	0.21733	ALPHLNR (4) unthinned	-0.703837	ALPHN (4) unthinned	-40.32688
41	ZETAR(5)	0.181586	ALPHLNR (5) unthinned	-0.739581	ALPHN (5) unthinned	-42.3749
42	ZETAR(6)	0.155856	ALPHLNR (6) unthinned	-0.765311	ALPHN (6) unthinned	-43.8491
43	ZETAR(7)	0.136473	ALPHLNR (7) unthinned	-0.784694	ALPHN (7) unthinned	-44.95966
44	ZETAR(8)	0.121357	ALPHLNR (8) unthinned	-0.79981	ALPHN (8) unthinned	-45.82576
45	ZETAR(9)	0.109243	ALPHLNR (9) unthinned	-0.811924	ALPHN (9) unthinned	-46.51981
46	ZETAR(10)	0.099322	ALPHLNR (10) unthinned	-0.821846	ALPHN (10) unthinned	-47.0883
47	ZETAR(11)	0.091047	ALPHLNR (11) unthinned	-0.83012	ALPHN (11) unthinned	-47.56238
48	ZETAR(12)	0.084043	ALPHLNR (12) unthinned	-0.837125	ALPHN (12) unthinned	-47.96372
49	ZETAR(13)	0.078037	ALPHLNR (13) unthinned	-0.843131	ALPHN (13) unthinned	-48.30783
50	ZETAR(14)	0.07283	ALPHLNR (14) unthinned	-0.848337	ALPHN (14) unthinned	-48.60612
51	ZETAR(15)	0.068275	ALPHLNR (15) unthinned	-0.852893	ALPHN (15) unthinned	-48.86716
52	ZETAR(16)	0.064254	ALPHLNR (16) unthinned	-0.856913	ALPHN (16) unthinned	-49.0975
53	ZETAR(17)	0.060681	ALPHLNR (17) unthinned	-0.860487	ALPHN (17) unthinned	-49.30226
54	ZETAR(18)	0.057483	ALPHLNR (18) unthinned	-0.863684	ALPHN (18) unthinned	-49.48546
55	ZETAR(19)	0.054605	ALPHLNR (19) unthinned	-0.866562	ALPHN (19) unthinned	-49.65035
56	ZETAR(20)	0.052002	ALPHLNR (20) unthinned	-0.869166	ALPHN (20) unthinned	-49.79952
57	ZETAR(21)	0.049635	ALPHLNR (21) unthinned	-0.871532	ALPHN (21) unthinned	-49.93513
58	ZETAR(22)	0.047474	ALPHLNR (22) unthinned	-0.873693	ALPHN (22) unthinned	-50.05894
59	ZETAR(23)	0.045493	ALPHLNR (23) unthinned	-0.875674	ALPHN (23) unthinned	-50.17242
60	ZETAR(24)	0.043671	ALPHLNR (24) unthinned	-0.877496	ALPHN (24) unthinned	-50.27683
61	ZETAR(25)	0.041989	ALPHLNR (25) unthinned	-0.879178	ALPHN (25) unthinned	-50.37319

pre Z37=ATAN(F18/(2*K8*S37));…; a pre Z61= ATAN(F18/(2*K8*S61))

pre AB37=AP35+Z37; …; a pre AP61=AP35+Z61

pre AD37=AB37/K9;…; a pre AD61=AB61/K9



Obrázok 3: Distribúcia elementálnych pracovných rezných uhlov pozdĺž lip regiónu (a) a pozdĺž chisel regiónu (b)

Obrázok 5(a) ukazuje, že normálové pracovné uhly, γ_n, majú negatívne hodnoty blízko fazetky (chisel edge) a narastajú do pozitívnych hodnôt pozdĺž hlavnej reznej hrany. Obrázok 5(b) ukazuje, že pre fazetku (chisel edge) pracovné rezné hrany sú negatívne od malých negatívnych hodnôt blízko osi nástroja narastajúcich do veľkých negatívnych hodnôt na rohu fazetky. Tlak tvorený fazetkou pri vŕtaní bude tvoriť veľkú väčšinu tlačnej sily, nie však krútiaceho momentu. Veľké hodnoty uhlu 2P spôsobia malý kladný nárast v hlavnom reznom regióne (lip), čo trochu zníži krútiaci moment. Na druhej strane ale fazetka (chisel edge) potlačí pracovné uhly do negatívnejších hodnôt, čo zvýši tlačnú silu. Keď berieme do úvahy celý vrták, potom zvyšovaním 2P hodnôt je možné znižovať celkový krútiaci moment pri malom náraste tlačnej sily. Zvyšovaním geometrického uhla ψ rozloženie pracovných uhlov na hlavnej reznej hrane zdvihne distribúciu uhlov do pozitívnejších hodnôt – pre rezanie lepších uhlov, takže sily súvisiace s tlakom a krútiacim momentom poklesnú. Model ukazuje, že konštantné hodnoty D, 2P, ψ a znižujúce sa údaje pre 2W/D vytvoria dlhšiu hlavnú reznú hranu a skrátia vedľajšiu reznú hranu, čo spôsobí, že distribúcia elementálnych pracovných uhloch na obidvoch rezných hranách sa zlepší (narastie do kladnejších alebo klesne do menej záporných hodnôt), čo je lepšie pre vŕtanie. Model ukazuje, že nárasty v hodnote δ_o uhla zvýšia normálové uhly pozdĺž lipového regiónu, čo zníži sily. Keďže tento uhol sám o sebe neovplyvňuje geometriu vrtáku na fazetke – chisel regióne tam rezné sily ostanú nezmenené. Rezný posun (f) nebude ovplyvňovať distribúciu pracovných uhlov na hlavnej reznej hrane (lip región), ale spôsobí, že negatívne pracovné uhly na fazetke poklesnú (vylepšia sa) s nárastom posuvu. Tlačná sila a krútiaci moment budú narastať približne lineárne s posuvom z dôsledku lineárneho narastania v ploche rezu. Ďalší druh modelovania umožní študovať distribúciu tlačných síl a krútiaceho momentu cez zmenu vstupných údajov a ich vplyvu na elementálnu distribúciu pracovných uhlov na rezných hranách vrtáku. Táto problematika je podrobnejšie diskutovaná v článkoch [1 – 4] s referenciou na zdroje [5 – 7].

Záver

Táto štúdia poukázala na možnosti použitia počítačov vo výskume a výučbe strojársky zameraných predmetov – konkrétne trieskového obrábania – vŕtaním. Analýza rezného stavu a skĺbenie statickej geometrie vrtákov s dynamickým správaním sa počas rezania na rôznych rezných rýchlostiach a posuvoch ukázali, že vzťahy sú zložité. Z toho dôvodu príprava tohoto programu a optimizačnej metódy pre potreby výučby a výskumu bola nevyhnutná. Praktické použitie metódy znížilo výrazne náklady na výskum a výučbu, odstránilo potrebu zložitého experimentálneho

testovania. Autor verí, že jeho výsledky zaujmú aj slovenských čitateľov a uvíta komunikáciu a prípadnú spoluprácu v tejto oblasti.

Literatúra

1. AUDY J. "*Exploring the Role of Computer Modelling and Image Analysis in Assessing Drill Design Features and Performance***",** Journal of Manufacturing Engineering, (Strojnicky Casopis), 57, Vol 6 December 2006, pp. 1-17.
2. AUDY J.: "*A Study of the Effect of Coatings on the Drill Life*", Manufacturing Engineering (Journal), Vyrobne Inzinierstvo, No. 1, Vol. VI, Technical University – Kosice, Slovakia, January 2007.
3. AUDY J.: "*A Study of the Effect of Variations in Drill Point Geometry on the Experimental and Predicted Cutting Forces Generated by Uncoated and Coated Drills*", Materials Engineering (Journal), ISSN 1335-0803, Vol. 13, No. 4, 2006, pp. 8 to 15. WWW informations-http://fstroj.utc.sk/journal-mi.
4. AUDY J., DOYLE E.D., AUDY K., HARRIS S. G.: "*A Study of the Effect of Drill Point Geometry and Physical Vapour Deposited Coatings on the Cutting Forces Generated by Twist Drills*", 5th Conference on "*Behaviour of Materials in Machining*", Chester, UK, 11-14 November, 2002
5. AUDY J.: "*The Influence of Hard Coatings on the Performance of Twist Drills*", Thesis for Master of Engineering Science in Research, The University of Melbourne, Australia, June 2002.
6. AUDY J.: "*Assessment of Metal Machining Process Parameters and the Development of Adaptive Control*", PhD Thesis, The University of South Australia, June 1996.
7. ARMAREGO E. J. A: "*Material Removal Processes - Twist Drills and Drilling Operations*", Manufacturing Science Group, Department of Mechanical and Manufacturing Engineering, The University of Melbourne, 1996

Poďakovanie

Autor článku ďakuje Edith Cowan Univerzite za podporu jeho vedeckých záujmov vo výskume a výučbe.

Dr. AUDY Jaromir
e-mail: j.audy@ecu.edu.au
Edith Cowan University
School of Enterprise and Technology
Robertson Drive
South West Campus Bunbury
Western Australia 6230

JEDNODUCHO – ZÁBAVA S DÁTAMI

Abstrakt: *príspevok sa zaoberá návrhom a realizáciou univerzálneho výkonného skriptu v jazyku PHP. Skript je určený na vyhodnotenie výsledkov prieskumu, ktoré sú uložené v skupine databázových tabuliek (MySQL). Navrhnutá metóda umožňuje riešenie hodnotenia výsledkov prieskumu zvlášť pre každú otázku dotazníka a tiež s rozkladom výsledkov podľa ďalších šiestich parametrov (teda ďalších otázok dotazníka). Funkčnosť metódy je ilustrovaná na prieskume pripravenosti študentov našej fakulty na štúdium informačných a komunikačných technológií.*

Kľúčové slová*: dotazník, výsledky prieskumu, PHP, MySQL, internet.*

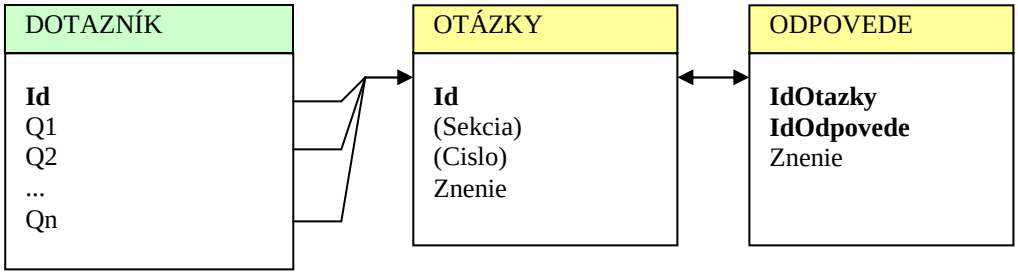
Úvod

Využívanie rôznych dotazníkov na zisťovanie informácií a názorov respondentov na určitú problematiku je štandardným a zaužívaným postupom v mnohých oblastiach. Postup má dve etapy – získavanie údajov a spracovanie údajov. Získanie údajov predstavuje určitý organizačný problém, či úlohu. Spracovanie údajov (najmä v prípade malých prieskumov) je pomerne namáhavé. V súčasnosti spracovanie obyčajne spočíva v prenesení výsledkov prieskumu do prostredia MS Excel a jeho následnom (manuálnom) spracovaní. Týmto spôsobom sme doteraz postupovali aj na fakulte BERG (baníctva, ekológie, riadenia a geotechnológií) Technickej univerzity v Košiciach, kde už niekoľko rokov sledujeme počítačovú gramotnosť a pripravenosť študentov na štúdium informačných a komunikačných technológií [2]. Prieskum je vykonávaný po ukončení prvého ročníka na odbore Riadenie procesov získavania a spracovania surovín. Prvýkrát (2001-02) bol prieskum realizovaný v klasickej papierovej forme, v nasledujúcich rokoch to už bola forma elektronická. Výsledky prieskumu boli ukladané do databázovej tabuľky (MySQL). Vytvorením výkonného (a pritom univerzálneho) skriptu (v jazyku PHP) pre účely vyhodnotenia prieskumu vznikla možnosť jednoduchého získavania nielen výsledkov prieskumu, ale aj rôznych informácií, závislostí a trendov, ktoré sa nachádzajú v údajoch prieskumu. Jednoducho – zábava s dátami.

Hodnotenie dotazníka

Predpoklady navrhnutej metódy

Metóda efektívneho hodnotenia výsledkov prieskumu vychádza zo štruktúry dátového modelu podľa Obr. 1.



Obr. 1 Štruktúra dátového modelu dotazníka

Uvedená štruktúra dátového modelu môže byť realizovaná dvoma spôsobmi – prvý je aplikácia uvedenej štruktúry na konkrétny tvar dotazníka (teda otázky a príslušné odpovede), ktorý je potrebné v tomto tvare zostaviť a naprogramovať (pre každý konkrétny prieskum vždy znovu) – veľmi pracný spôsob. Jeho výhodou je možnosť detailného prispôsobenia požiadavkám a jednotlivým otázkam, prípadne odpovediam. Druhým spôsobom je realizácia interaktívneho vývojového prostredia (IVP) pre tvorbu elektronického dotazníka v ľubovoľnej oblasti s tým, že toto vybavenie je založené na využití uvedeného dátového modelu. Takéto IVP bolo na našom pracovisku zostavené [3] (skriptovací jazyk PHP a databázový systém MySQL) a je vo viacerých projektoch úspešne využívané.

IVP umožňuje dosiahnuť prenesenie celej tvorby dotazníka – návrh sekcií, znenia a typov otázok, jednotlivých odpovedí ako aj všetkých opráv a doplnkov priamo na zadávateľa. Vytvorenie dotazníka sa tak podstatne zjednoduší a súčasne aj urýchli. IVP zakrýva pred používateľom veľa špecializovaných technologických problémov, postupov aj riešení z oblasti databázových systémov, z oblasti technológie html, interaktívnych skriptov php, kaskádových štýlov css ako aj ich vzájomnej komunikácie a umožňuje mu tak sústrediť sa na obsahovú stránku elektronického dotazníka.

Hlavná funkcia IVP – podpora tvorby elektronického dotazníka – je riešená v multijazykovom prevedení. To znamená, že umožňuje vytvorenie (rovnakých) otázok a odpovedí na nich vo viacerých jazykoch. Medzi čiastkové funkcie IVP patrí pridanie ďalšej otázky, zoznamu alebo možností (typov) odpovedí na zadanú otázku, pridanie novej sekcie a pridanie popisu. Ďalej sem patrí editovanie alebo zrušenie týchto prvkov dotazníka, ako aj funkcie pre úpravu vzhľadu dotazníka. Súčasťou IVP je tiež skupina funkcií pre štandardné vyhodnocovanie dotazníka. Táto skupina umožňuje prispôsobenie spôsobu vyhodnotenia na konkrétne podmienky príslušného projektu. Významnou črtou IVP je možnosť diaľkovej správy a vyhodnocovania projektu (to znamená, že uloženie projektu a jeho databázy je prakticky ľubovoľné v rámci internetu). Okrem tvorby dotazníka IVP podporuje vytvorenie elektronického testu a elektronickej ankety ako špeciálnych typov elektronického formulára.

Činnosť IVP je založená na komunikácii s používateľom prostredníctvom webovej stránky s niekoľkými rôznymi formulármi, cez ktoré sú zadávané jednotlivé prvky elektronického dotazníka. Po vyplnení a odoslaní sa dotazník spracuje a zadané údaje (týka sa štruktúry, sekcií, otázok a možných odpovedí na nich) sa uložia do odpovedajúcich databázových tabuliek.

IVP umožňuje vytvárať viac (celú skupinu) elektronických formulárov. Preto sa každý tvorca elektronického formulára musí prihlásiť do systému IVP (login a heslo). Vytváracie aj vyhodnocovacie tabuľky formulára obsahujú vo svojom názve aj príslušný login, čím sú od seba jednotlivé projekty formulárov dostatočne odlíšené. Vhodným výberom z uvedených tabuliek je následne vygenerovaná stránka elektronického dotazníka.

Princíp metódy hodnotenia

Navrhnutá metóda umožňuje riešenie hodnotenia výsledkov prieskumu zvlášť pre každú otázku dotazníka a tiež s rozkladom výsledkov podľa ďalších štyroch parametrov (teda ďalších otázok dotazníka). Riešenie pozostáva z dvoch krokov.

Prvým krokom je výber kombinácie otázok dotazníka, vzájomná súvislosť ktorých používateľa zaujíma. Používateľ má na tento účel k dispozícii formulár s piatimi vstupnými prvkami v tvare zoznamov. Každý vstupný prvok je naplnený zoznamom všetkých otázok dotazníka, pričom v poradí prvá pozícia (implicitná voľba) je prázdna. Prázdna pozícia sa do procesu spracovania prenáša ako nulová hodnota. Prvý vstupný prvok neobsahuje prvú prázdnu pozíciu. Prvý vstupný prvok s prázdnou prvou pozíciou ukončuje zadanú kombináciu otázok dotazníka. Vstupný formulár okrem toho umožňuje výber oddeľovača desatinnej časti výsledkov (bodka, čiarka).

Druhým krokom je zistenie počtu všetkých odpovedí pre zvolenú kombináciu otázok, ich prepočet na percentá a výstup v tvare tabuľky. Tento proces má toľko úrovní (poddotazov), koľko vstupných prvkov má zvolenú neprázdnu hodnotu. Poddotazy nie sú v MySQL podporované (do verzie 4.1), preto je výpočet riešený formou vložených úrovní. Algoritmus výpočtu je znázornený časťou kódu skriptu v jazyku php pre jednu (prvú) úroveň. Začína sa určením celkového počtu respondentov (premenná $pocet), nasleduje určenie názvu vybranej otázky ($n) pomocou výberového kritéria ($vyber0) a úvodná časť sa končí špecifikáciou výberového atribútu ($hlp) tabuľky dotazníka. Potom nasleduje určenie celkového počtu jednotlivých odpovedí na zvolenú otázku, ich prepočet na percentá a výpis v tvare tabuľky. Tabuľka má dva stĺpce, ktoré obsahujú názov odpovede a jej percentuálny podiel. V prípade viacerých úrovní sa po určení a výpise hodnôt odpovedí pre prvý parameter ($vyber0) vykonáva ten istý postup pre druhú úroveň ($vyber1) atď.

```
$sql="select count(Id) from ".$tbd;              //$tb – tabulka Otazky
$result = mysql_query($sql);
$poceta=mysql_fetch_array($result);
$pocet=$poceta[0];                               //celkovy pocet zaznamov
$id="Id";                                        //kluc tabulky Dotaznik ($tbd)
$co="Znenie_SVK";   //nazov atributu so znenim otazky – tabulka Odpovede ($tb1)
   $sql="select $co from $tb where ID=$vyber0";
   $result = mysql_query($sql);
   $poceta=mysql_fetch_array($result);
   $n=$poceta[0];                                //nazov vybranej otazky

$hlp='Q'.$vyber0;                                //nazov atributu tabulka Dotaznik
if($vyber0 && !$vyber1 && !$vyber2 && !$vyber3 && !$vyber4){        //len 1. vyber
   $q1="select $co, count($id) from $tb1,$tbd";
   $q1.=" where $tb1.IdOdpovede = $tbd.$hlp and IdOtazky = ".$vyber0." group by $co";
   @$r = mysql_query($q1);
   print "<table border='1'>";
   print "<tr align='center'><td><b>".$n."</b></td><td>%</td></tr>";
   while ($l=mysql_fetch_array($r)){
      $perc=$l[1]/$pocet*100;
      $cpocet1+=$perc;
      print "<tr><td>".$l[0]."</td><td align='right'>".$perc."</td></tr>";

   //priestor na vybery podla dalsich neprazdnych vstupnych prvkov

   }
   if($cpocet1<100.){                            //kontrola konzistencie dat dotaznika
      print "<tr><td> Spolu ($hlp)</td><td>".$cpocet1."</td></tr></table>";
   }
   else
      print "<tr><td> Spolu</td><td>".$cpocet1."</td></tr></table>";
   mysql_free_result($r);
}
```

Podmienkou správnej činnosti metódy je konzistentnosť vstupných údajov. Na túto skutočnosť je potrebné pamätať už vo fáze realizácie dotazníka (každý vstupný prvok má mať predvolenú hodnotu – selected). V prípade, že niektorá otázka dotazníka umožnila nekorektný vstup (obyčajne nezadanie žiadnej hodnoty), prejaví sa táto skutočnosť v hodnotení výsledkov celkovou sumou menšou ako 100. V takejto situácii sa pri otázke označí aj identifikátor „chybného" stĺpca tabuľky. Na základe toho je možné vykonať vhodnú korekciu (buď doplniť chýbajúci údaj, alebo zrušiť celý riadok, teda jedného respondenta).

Realizácia metódy

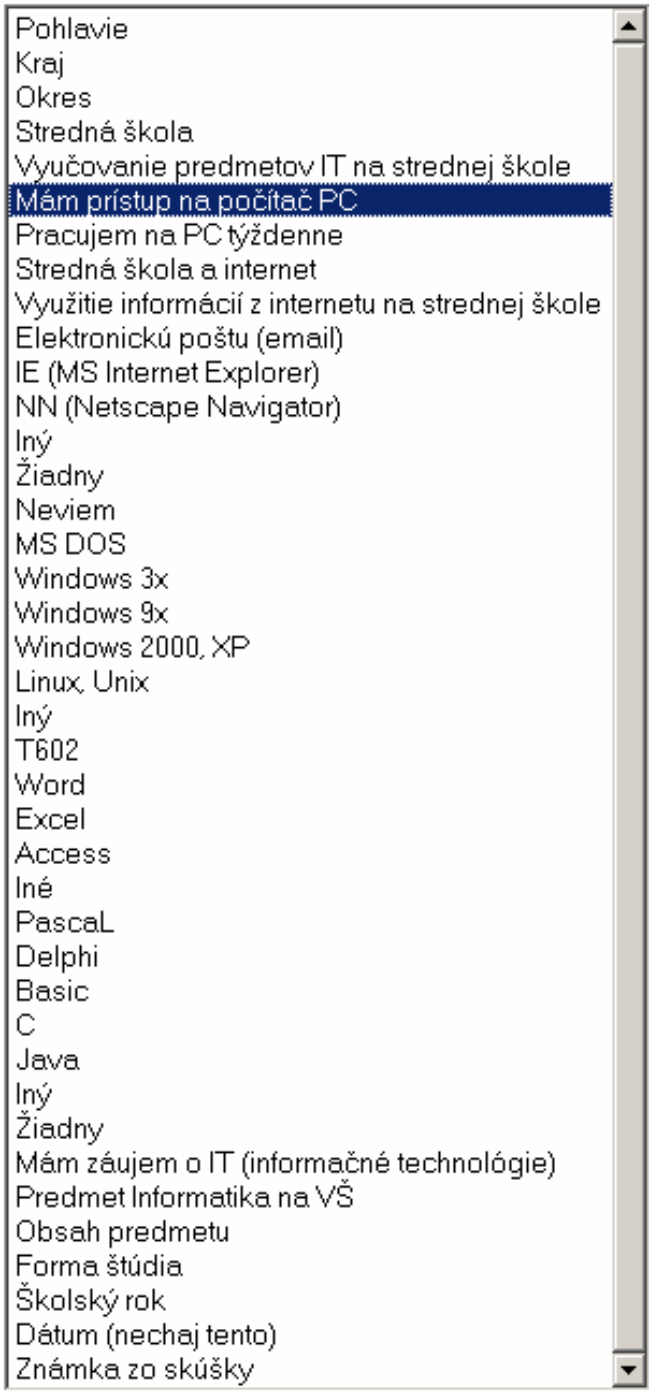
Metóda hodnotenia výsledkov prieskumu je realizovaná vo forme dvoch súborov. Jeden súbor realizuje algoritmus niekoľkonásobných výberov, ich spracovanie a výpis v tvare tabuľky. Druhý súbor (db.php) zabezpečuje pripojenie na príslušnú databázu (autentifikácia), prácu s odpovedajúcimi tabuľkami a stĺpcami týchto tabuliek. Druhý súbor je implicitnou súčasťou súboru prvého (require).V prípade viacjazyčného prevedenia elektronického dotazníka stačí zmeniť identifikáciu jazyka (namiesto Znenie_SVK bude Znenie_ENG) v označení atribútu tabuľky.

Skript metódy má modulárnu štruktúru. Je tvorený celkom 9 funkciami, kombináciou ktorých je realizovaný algoritmus. Päť funkcií má pomocný charakter (formátovanie číselného výstupu f(), tlač hlavičky tbhd()a riadku tabuľky tbln2(), celkovej sumy tbsp() a vyplnenie prázdnych polí tabuľky sp() –   – pre prehliadač Internet Explorer). Funkcia otazka() vracia názov vybranej otázky, alebo celkový počet záznamov. Funkcie vypis() a vypis1() zabezpečujú výstup príslušných výberov v tabuľkovom tvare (pozri ukážku skriptu). Funkcia selcr() zabezpečuje vytvorenie vhodného príkazu select pre požadovaný počet úrovní.

K realizácii metódy sú možné dva prístupy. **Prvý prístup** vytvára n vstupných zoznamov (v našom prípade n=6). **Druhý prístup** vytvára jeden vstupný zoznam otázok s možnosťou výberu viacerých (max. n=6) otázok (Ctrl C). Z hľadiska potrebnej plochy okna je výhodnejší (úspornejší) druhý prístup. Výhodou prvého prístupu je možnosť voľby ľubovoľného poradia otázok, čo pri druhom prístupe nie je možné (tam je poradie dané usporiadaním otázok v dotazníku).

Príklad využitia - prieskum počítačovej gramotnosti na F BERG

Dotazník je tvorený troma sekciami otázok. Otázky prvej sekcie špecifikujú respondentov z hľadiska pohlavia, bydliska, možnosti prístupu na počítač a typu strednej školy. Pretože v našom prípade je prieskum zameraný aj na hodnotenie poznatkov z oblasti informačných technológií získaných na strednej škole, druhá sekcia je venovaná tejto problematike. Otázky sú zamerané na dĺžku výučby predmetov z oblasti informačných technológií, aktívnu prácu s počítačom, používanie elektronickej pošty, znalosť používania operačného systému a znalosť programovacieho jazyka, vybavenosť strednej školy internetom ako aj na znalosti a využívanie internetových služieb. V tretej sekcii je hodnotený záujem o štúdium informačných technológií na vysokej škole, názor poslucháčov na potrebu výučby predmetu informatika a tiež dosiahnutá známka z tohto predmetu. Otázky dotazníka ilustruje Obr. 2.



Obr. 2 Zoznam otázok dotazníka FBERG

Niekoľko výsledkov

Funkčnosť a možnosti navrhnutej metódy hodnotenia výsledkov prieskumu demonštruje niekoľko výsledkov prieskumu počítačovej gramotnosti na FBERG v akademickom roku 2004-05. Postupne sú v tabuľkách uvedené výsledky prístupu na počítač, pohlavie – tab.1, využívanie elektronickej pošty – tab. 2 a forma štúdia – tab.3. Nakoniec je uvedené porovnanie prístupu poslucháčov na počítač za posledné tri akademické roky – tab. 4.

Tabuľka 1 – Prístup na počítač a prístup na počítač podľa pohlavia

Mám prístup na počítač PC	%
Doma	91.58
Inde	6.32
Nemám	2.11
Spolu	100.00

Mám prístup na počítač PC	%	Pohlavie	%
Doma	91.58	Muž	45.26
		Žena	46.32
Inde	6.32	Muž	3.16
		Žena	3.16
Nemám	2.11	Muž	2.11
Spolu	100.00		100.00

Tabuľka 2 – Prístup na počítač podľa pohlavia a využívania e-mailu

Mám prístup na počítač PC	%	Pohlavie	%	Elektronickú poštu (email)	%
Doma	91.58	Muž	45.26	nevyužívam	4.21
				využívam	36.84
				využívam zriedka	4.21
		Žena	46.32	nevyužívam	3.16
				využívam	36.84
				využívam zriedka	6.32
Inde	6.32	Muž	3.16	nevyužívam	1.05
				využívam	1.05
				využívam zriedka	1.05
		Žena	3.16	využívam	3.16
Nemám	2.11	Muž	2.11	využívam	2.11
Spolu	100.00		100.00		100.00

Tabuľka 3 – Prístup na počítač podľa pohlavia, využívanie e-mailu a forma štúdia

Mám prístup na počítač PC	%	Pohlavie	%	Elektronickú poštu (email)	%	Forma štúdia	%
doma	91.58	muž	45.26	nevyužívam	4.21	denná	3.16
						externá	1.05
				využívam	36.84	denná	32.63
						externá	4.21
				využívam zriedka	4.21	denná	4.21
		žena	46.32	nevyužívam	3.16	denná	3.16
				využívam	36.84	denná	30.53

Mám prístup na počítač PC	%	Pohlavie	%	Elektronickú poštu (email)	%	Forma štúdia	%
						externá	6.32
				využívam zriedka	6.32	denná	5.26
						externá	1.05
inde	6.32	muž	3.16	nevyužívam	1.05	denná	1.05
				využívam	1.05	denná	1.05
				využívam zriedka	1.05	denná	1.05
		žena	3.16	využívam	3.16	denná	1.05
						externá	2.11
nemám	2.11	muž	2.11	využívam	2.11	denná	1.05
						externá	1.05
spolu	100.00		100.00		100.00		100.00

Tabuľka 1 – Vývoj prístupu na počítač za tri roky

Mám prístup na počítač PC	02-03 %	03-04 %	04-05 %
Doma	66.67	84.43	91.58
Inde	21.28	8.20	6.32
Nemám	12.06	7.38	2.11
Spolu	100.00	100.00	100.00

Záver

Navrhnutý a realizovaný algoritmus umožňuje veľmi rýchle, jednoduché a pohodlné získanie výsledkov prieskumu až do šiestich úrovní. Požadované závislosti stačí zvoliť vo vstupnom formulári a okamžite je k dispozícii výsledok vo forme tabuľky. V prípade potreby je možné uvedený algoritmus rozšíriť aj na (ľubovoľný) väčší počet úrovní. Vzhľadom na štandardné (a pomerne pracné) postupy spracovania výsledkov v prostredí MS Excel poskytuje využitie uvedenej metódy významnú úsporu času, zjednodušenie získania výsledkov najmä pre vyšší počet úrovní a tiež vylúčenie možných chybových stavov (pri manuálnych postupoch v MS Excel). Za účelom jednoduchého prenosu výsledkov do prostredia tabuľkového kalkulátora (MS Excel alebo OO Calc - Open Office) je riešená aj možnosť voľby oddeľovača desatinnej časti (bodka alebo čiarka). Sekundárnou funkciou algoritmu je možnosť kontroly konzistencie vstupných údajov. Nekonzistentnosť sa prejaví sumárnou percentuálnou hodnotou menšou ako 100. Ďalší postup úpravy vstupných údajov závisí potom na rozhodnutí vyhodnocovateľa. Využitie zostaveného skriptu umožňuje stanovenie trendov, ak máme k dispozícii výsledky prieskumu za niekoľko časových období (napr. akademických rokov).

Zostavená dvojica skriptov je bezplatne k dispozícii všetkým záujemcom na adrese autora.

Literatúra

[1] HOROVČÁK, P.: Prieskum pripravenosti študentov na štúdium informačných technológií. Informatika v škole č. 26, 2003, ÚIPŠ Bratislava, ISSN 1335-616X, str. 10 - 19

[2] HOROVČÁK P.: Pripravenosť študentov F BERG na štúdium informačných technológií. Acta Montanistica Slovaca, 4, 2002. ISSN 1335 – 1788, pp. 306 – 310

[3] HOROVČÁK, P., OROSZ M. 2004: Dotazník na webe – ako na to?. Informatika v škole č. 27, 2004, ÚIPŠ Bratislava, ISSN 1335-616X, str. 11 – 19

[4] LEVENSTEIN, Aaron Quote [online] [27.7.2005] available from <http://www.global-investor.com/quote/5431/Aaron-Levenstein>

Poznámka: Príspevok bol riešený v rámci projektov KEGA 3/3084/05 (H), KEGA 3/3125/05 (M), KEGA 1/3126/05 (B), VEGA 1/2179 /05 (D), VEGA 1/2160 /05 (K) a ABILITIES Co 027306 (6RP)

Pavel HOROVČÁK

Ústav riadenia a informatizácie procesov
Fakulta BERG, Technická univerzita v Košiciach
tel. 421 55 602 5176
fax 421 55 6339772
e-mail Pavel.Horovcak@tuke.sk