**Բովանդակություն**

[Ներածություն 3](#_Toc481680704)

[Խնդիր դրվածքը 4](#_Toc481680705)

[**Գլուխ 1 Տեսական դրույթներ** 5](#_Toc481680706)

[1.1 Ինտեգրալ սխեմաների թեստավորման մեթոդները 6](#_Toc481680707)

[1.2 Տվյալների աբստրակտ տիպերը 11](#_Toc481680708)

[1.3 Տվյալների հետ կատարվող գործողությունները 12](#_Toc481680709)

[**1.4 QT գրադարանի տարրերը** 26](#_Toc481680710)

[Գլուխ 2 Ծրագրային իրագործում 28](#_Toc481680711)

[2.1 Տվյալների պահպանման ինտերֆեյսները 29](#_Toc481680712)

[2.2 Տվյալների խմբավորում 31](#_Toc481680713)

[2.3 Տվյալների քեշավորում 31](#_Toc481680714)

[2.4 Սփռված տվյալների հաջորդական ներկայացում 32](#_Toc481680715)

[2.5 Պայմանին համապատասխան տվյալների վերադարձ 33](#_Toc481680716)

[**Գլուխ 3 Տնտեսագիության բաժին** 35](#_Toc481680717)

[3.1 Ներածություն, խնդրի դրվածքը 35](#_Toc481680718)

[3.2 Թեման մշակող ձեռնարկության նկարագիրը 35](#_Toc481680719)

[3.3 Թեմայի մշակման գործընթացի կազմակերպումը: Թեմայի կառուցվածքը Թեման իրականացնող անձնակազմի ձևավորումը 36](#_Toc481680720)

[3.4 Աշխատանքների էտապավորումը եվ օրացուցային պլանի կազմումը, աշխատատարությունների որոշումը 36](#_Toc481680721)

[3.5 Թեմայի մշակման ծախսերի նախահաշվի կազմումը եվ վերլուծությունը 38](#_Toc481680722)

[3.5.1 Նյութեր, գնովի պատրաստվածքներ և կիսաֆաբրիկատներ 39](#_Toc481680723)

[3.5.2 Հատուկ սարքավորումներ գիտական, փորձարարական աշխատանքների համար 40](#_Toc481680724)

[3.5.3 Հիմնական այլ միջոցներ 41](#_Toc481680725)

[3.5.4 Գիտաարտադրական անձնակազմի աշխատավարձեր 41](#_Toc481680726)

[3.5.5 Սոցիալապահովագրական հատկացումներ 42](#_Toc481680727)

[3.5.6 Գիտաարտադրական գործուղումներ 42](#_Toc481680728)

[3.5.7 Կողմնակի կազմակերպություններ, աշխատանքներ և ծառայություններ (հատուկ նպատակային ծախսեր) 43](#_Toc481680729)

[3.5.8 Այլ ուղղակի ծախսեր 43](#_Toc481680730)

[3.5.9 Վերադիր ծախսեր 43](#_Toc481680731)

[**Գլուխ 4 Կենսագործունեության անվտանգության բաժին** 46](#_Toc481680732)

[4.1 Հաշվիչ կենտրոններում աշխատանքի պաշտպանությանը ներկայացվող պահանջները 46](#_Toc481680733)

[**Գլուխ 5 Բնապահպանության բաժին** 52](#_Toc481680734)

[5․1 Արհեստական աղբյուր ունեցող էլեկտրամագնիսական դաշտերը և նրանց ազդեցությունը շրջակա միջավայրի վրա 52](#_Toc481680735)

[Եզրակացություն 57](#_Toc481680736)

[Օգտագործված գրականության ցանկ 58](#_Toc481680737)

Ներածություն

Ինտեգրալ սխեմաների թեստավորումը կատարվում է տարբեր կառուցվածքների և տեխնոլոգիաների միջոցով՝ կախված ինտեգրալ սխեմայի բարդությունից և տեսակից, որոնց ընթացքում կուտակվում են ահռելի քանակությամբ տվյալներ:Թեստավորումը թույլ է տալիս ԻՍ-ների հետ իրականացնել ՝

* Միկրոսխեմայի ֆունկցիանալ վերահսկողություն
* Դեռևս տպասալում գտնվող միկրոսխեմայում առաջացող թերությունների հայտնաբերում
* Սկավառակում գտնվող միկրոսխեմայի էլեկտրական պարամետրերի չափումներ

Թեստավորման արդյունքներից կարելի է այնպիսի ինֆորմացիա ստանալ, ինչպիսիք են մակերևույթի վիճակը, վնասված կամ ոչ լրիվ փորագրումը, օքսիդի շերտի ոչ բավարար հաստությունը, անցման ճշգրտությունը և այլն։ Որպեսզի ԻՍ-ի թեստավորումը հարմար լինի, վերահսկող տարրերը էլեկտրական շղթային միացվում են հաջորդաբար կամ զուգահեռ: Բացի թեստավորող միկրոսխեմաներից, առանձին վերցված կոմպոնենտների, օրինակ, տրանզիստորների կամ դիոդների վերահսկողությունը կարելի է իրականացնել թեստավորող բյուրեղների միջոցով:

Թեստավորման գործընթացի ժամանակ գեներացված տվյալները բնականաբար պետք է ինչ-որ կերպ պահվեն և ինժեներներին հնարավորություն ընձեռվի աշխատել այդ տվյալների հետ, անել համապատասխան եզրակացություններ: Այս ծրագրի գլխավոր խնդիրն է կարողանալ պահպանել այդ տվյալները, որոշակի հաշվարկներ իրականացնել դրանց հետ և տարբեր գրաֆիկական լուծումներով թեստի արդյունքները ներկայացնել ինժեներին: Խնդրի լուծման ընթացքում մշակել և իրականցվել են թեստավորման արդյունքների հետ աշխատող ինտերֆեյսներ, որոնցից օգտվելու են ծրագրի մնացած մոդուլները և նախագծվել են տվյալների պահման արդյունավետ կառուցվածքներ։

# Խնդիր դրվածքը

Մշակել թեստավորման արդյունքների հետ աշխատող ինտերֆեյսներ, իրականացնել այդ ինտերֆեյսները և նախագծել հիշողության մեջ արդյունավետ պահման կառուցվածքներ:

Պահման կառուցվածքները պետք է ապահովեն՝

* Հիշողության օպտիմալ օգտագործում
* Հիշողության մեջ սփռված տվյալների հաջորդական ներկայացում
* Պահվող տվյալների հետ կատարվող գործողությունների արագագործություն (փնտրում, ավելացում, կարգավորում)
* Պայմանին համապատասխան տվյալների արագ վերադարձ

Գլուխ 1   
 Տեսական դրույթներ

# Ինտեգրալ սխեմաների թեստավորման մեթոդները

Ինտեգրալ սխեմաների թեստավորումը կատարվում է տարբեր կառուցվածքների և տեխնոլոգիաների միջոցով՝ կախված ինտեգրալ սխեմայի բարդությունից և տեսակից: Հիմնականում գոյություն ունեն երեք տեսակի թեստավորման մեթոդ՝

* Օպերատիվ վերահսկողության մեթոդ
* Վիզուալ վերահսկողության մեթոդ
* Տպասալի վրա թեստավորող սարքավորումների օգտագործման մեթոդ

Օպերատիվ վերահսկողության մեթոդ

Ինտեգրալ սխեմաների հետ արտադրական գործընթացներից հետո, ինչպիսիք են էպիտեկսիան և դիֆուզիան, անհրաժեշտ է իրականացնել ժապավենների, p-n անցումների խորությունների և կոնցենտրացիաների մակերևույթների վերաչափումներ։ Այդ չափումները կատարվում են առանձին հսկողությամբ։

Վիզուալ վերահսկողության մեթոդ

Տվյալների մեծ քանակ տպասալերի հետ կապված կարելի է ստանալ վիզուալ վերահսկողության մեթոդի շնորհիվ, որի օգնությամբ կարելի է ճանաչել այնպիսի ցուցանիշներ,ինչպիսիք են մակերևույթի վիճակը, վնասված կամ ոչ լրիվ փորագրումը, օքսիդի շերտի ոչ բավարար հաստությունը, անցման ճշգրտությունը և այլն։ Վիզուալ վերահսկողության մեթոդը հավանաբար հանդիսանում է ամենատրիվիալ մեթոդը վերջին երեք թվարկված մեթոդներից, բայց հաշվի չառնելով իր պարզությունը, մեծ դեր է խաղում ինտեգրալ սխեմաների արտադրության և թեստավորման ոլորտում։ Այս մեթոդը թույլ է տալիս կատարել տպասալի ուսումնասիրություն մանրադիտակով,շատ մեծ խոշորացմամբ (80ից-400 անգամ) և օգտագործել բազում վիզուալիզացիայի մեթոդներ թերմոգրաֆիայի և նմանատիպ այլ գործընթացների համար։ Գոյություն ունեն բազում խոտաններ, որոնցով կարող են օժտված լինեն պատրաստի տպասալերը։ Ամենալուրջ խոտանը հանդիսանում է օքսիդի շերտի ծակոտկենությունը, որը հեշտ կարելի է հայտնաբերել վիզուալ վերահսկողության մեթոդի ժամանակ մանրադիտակով միկրոսխեման դիտարկելիս։ Օքսիդի շերտի ծակոտկենությունը դա օքսիդի շերտի փոքրիկ անցքերն են, որոնք հիմնականում առաջանում են փոշու կամ լուսային կաղապարի անսարքության հետեվանքով։ Եթե դիտարկվող խոտանը հայտնաբերվում է տպասալի ծայրահեղ կետում, ապա հաջորդող դիֆուզիան կարող է հանգեցնել անցումային պրոցեսի փակման, ինչը կարող է շարքից հանել ամբողջ միկրոսխեման։

Վիզուալ վերահսկողության մեթոդի արդյունավետ իրականացումը հանդիսանում է սկանավորող էլեկտրոնային մանրադիտակի օգտագործումը, որի միջոցով օպերատորը կարող է դիտարկել ինտեգրալ սխեմայի տեղագրական և էլեկտրական ռելիեֆը։ Այս մեթոդը թույլ է տալիս հեշտությամբ հայտնաբերել անցումների աղտոտվածությունը, փոշու տարբեր հատիկները, օքսիդի շերտի անցքերը, մետաղական բարակ շերտի վնասվածությունները։ Ջերմային փորձերի ժամանակ գերտաքացած տիրույթների բացահայտման համար տպասալի վրա գոյություն ունի ինֆրակարմիր սկանավորող միկրոսկոպ, որը բաղկացած է հատուկ ձայնագրող սարքին միացած ԻՍ-դետեկտորից: Այս սարքավորումը օգտագործվում է տպասալի կառուցվածքի որակի գնահատման համար՝ ջերմության և հզորության տարածման տեսանկյունից:

Տպասալի վրա թեստավորող սարքավորումների օգտագործման մեթոդ

Ինտեգրալ սխեման բաղկացած է մեծ թվով բարդ տարրերից, որոնք միավորված են մետաղական մայրուղիների միջև, որոնց հատումները հասնում են մի քանի հարյուրի կամ հազարի: Բացի այդ, տպասալն ունի շատ մեծ քանակությամբ անցումներ մի շերտից մյուսը (կոնտակտային պատուհաններ), ելուստներ ակտիվ և պասիվ տարրերի համար, բազմաթիվ կոնտակտային հրապարակներ և այլն: Տարրերի այսքան բարդ դասավորվածության պատճառով գործնականում անհնար է ամբողջովին վերահսկել բոլոր տարրերը՝ նրանց աշխատանքի մեծ ծավալի պատճառով: Չնայած այս ամենին, վերահսկողության անհրաժեշտությունը ակնհայտ է, հատկապես ԻՍ արտադրության տեխնոլոգիաների կատարելագործման փուլում: Էլեկտրական պարամետրերի և տեխնոլոգիական օպերացիաների իրականացման որակի վերահսկողությունը իրականացվում է հատուկ միկրոսխեմաների միջոցով, որոնք տեղադրված են տպասալի վրա՝ աշխատանքային բյուրեղների հետ: Թեստավորող միկրոսխեման պատրաստվում է նույն եղանակով, ինչ տպասալի վրայի սովորական միկրոսխեման, այն պարունակում է հատուկ համադրությամբ բոլոր բաղկացուցիչ տարրերը, որոնք թույլ են տալիս գնահատել ԻՍ-ի պարամետրերը և տեխնոլոգիական օպերացիաների իրականացման որակը: Որպեսզի ԻՍ-ի թեստավորումը հարմար լինի, վերահսկող տարրերը էլեկտրական շղթային միացվում են հաջորդաբար կամ զուգահեռ: Բացի թեստավորող միկրոսխեմաներից, առանձին վերցված կոմպոնենտների, օրինակ, տրանզիստորների կամ դիոդների վերահսկողությունը կարելի է իրականացնել թեստավորող բյուրեղների միջոցով: Տվյալ բյուրեղը պարունակում է իր մեջ իրարից մեկուսացված որոշակի տարրեր, որոնք գտնվում են ինտեգրալ սխեմայում: Թեստավորող բյուրեղի չափերը համեմատական են տպասալում ինտեգրալ սխեմայի չափերի հետ: Այսպիսի տեխնոլոգիայի օգտագործումը թույլ է տալիս իրականացնել բարձր տեխնոլոգիական վերահսկողություն ԻՍ-ի արտադրության վրա և նվազեցնել տպասալի թեստավորման փորձարկումների վրա ծախսվող ժամանակը և աշխատանքի ծավալը: Սիլիցիումի տպասալի վրա թեստավորող շերտերը գտնվում են աշխատանքային միկրոսխեմաների միջև: Տպասալի վրայի միկրոսխեմայի թեստավորումից առաջ, առաջին հերթին անհրաժեշտ է վերահսկողություն իրականացնել թեստավորորղ շերտերի վրա, հետո այդ վերահսկողության հաջղ իրականացումից հետո կարելի անցնել միկորսխեմայի չափումներին: Այս հերթականությունն ունի մի քանի առավելություն՝ ավելի հասանելի կոնտակտավորում առանձին թեստավորող տարրերի համար և լազերային եղանակով տարրերի առանձնացման ժամանակ միկրոսխեմայի վնասման հավանականության նվազում:

Այս ծրագրի մուտքին տրվում է csv ֆորմատի ֆայլ,որի պարունակությունն է՝   
 lot-ի համարը, wafer-ի համարը, device-ը, DieX, DieY, parameter-ը(չափվող պարամետրի արժեք), unit(չափվող պարամետրի միավոր), test pass(true կամ false), parameter limits(չափվող պարամետրի համար նախապես հայտնի միջակայք), temperature(այն ջերմաստիճանը, որում կատարվել են չափումները) և այլն :

Ծրագիրը հնարավորություն է տալիս ընտրել վերլուծության(analysis) տեսակը՝ Bin Trend, HBin Pareto, Wafer, Test Result by Lot/Wafer, Parameter Histogram, Parameter Probability, Summary Table և դիտարկել արդյունքները ընտրված գրաֆիկների կամ աղյուսակների միջոցով։

Արդեն ըստ այս ներկայացման ձևերի նախագծողը կարող է պատկերացում կազմել թեստի մասին, հասկանալ որ սարքերն են պիտանի թողարկման համար, որոնք փոփոխման կարիք ունեն, որոնք պետք է առհասարակ հեռացնել :

Ծրագիր մուտքին տրվելու է CSV(Comma-separated values) ֆորմատի ֆայլ, որի բովանդակությունը մոտավորապես այսպիսին է ՝

<File Signature>

Version: <>

Type: <File Data Type>

Info: <…>

@Context

Lot: <Lot name>

Wafer: <Wafer number>

Device: <Devise name>

@TestConditions

Temperature: 30

Corener: <…>

@Data

<Column1:Type>;<Column2:Type>;<Column3:Type>;…

Value;Value;Value;…

Value;Value;Value;…

Value;Value;Value;…

DataType ցույց է տալիս ֆայլի մեջ նկարագրված ինֆորմացիայի տիպը։

Lot, Wafer, Device համապատասխանաբար Lot-ի,Wafer-ի, Device-ի համարներն են։

TestConditions-ում նշվում են միջավայրի այն պայմանները, որոնց դեպքում կատարվել են չափումները, օրինակ՝ ջերմաստիճան։

TestProgram՝ կատարվող թեստերի խումբն է։

Եթե տրված է TestProgram-ն, ապա տրված է ամբողջ պրոցեսի ընթացքում կատարվող թեստերի հերթականությունը։

Data՝ չափման արդյունքում ստացված տվյալներն են։ Տվյալները ստորակետներով են բաժանված, որում գրված է տվյալ սյունակի անունը և դրանում պահվող տվյալների տիպը, որին հաջորդում են արժեքները ;-ով առանձնացված։

Պահանջվող վերլուծությունները կատարելուց հետո, ըստ այդ արդյունքների չիփերը բաշխվում են տարբեր արկղերում ( Bin )։ Նույն Bin-ի մեջ գտնվում են իրար նման die-երը, օրինակ նույն դեֆեկտը ունեցող die-եր։ Կարող են լինել տարբեր TestProgram-ներ և դրանցից կախված Bin-ի նշանակությունը կարող է փոխվել։ Bin Definition-ն և Bin Data ֆայլերը ներկայացնում են Bin-ի անունը, Bin-ի տեսակը։ BinType-ը կարող է լինել SBin կամ HBin։ Bin Definition-ը պարունակում է PassFail, որը կարող է լինել true կամ false, այն ցույց է տալիս, թե տվյալ Bin-ը լավն է, թե ոչ։ Կարող է լինել մեկ կամ ավելի Bin Definition և Bin Data ֆայլեր։ Bin Definition-ը կլինի մեկից ավելի, եթե կան տարբեր TestProgram-ներ։ Ծրագիրը Bin Data-ից ստանում է Bin Summary ֆայլ, որը ցույց է տալիս, թե որ Bin-ից քանի հատ կա Wafer-ի վրա։ Bin Data-ի հիման վրա կառուցվում են Bin Pareto գրաֆիկներ։

Չափումները կատարվում են ըստ Parameter տվյալների։ Նշվում է, թե ինչ parameter է չափվելու, թեստի համարը, չափման միավորը, չափված արժեքը, նաև նշվում է տվյալ արժեքը pass է, թե fail, որը որոշում է ըստ Parameter Limit-ների։ Չափման արդյունքը պարզապես թիվ է, դրանցից յուրաքանչյուրի համար նախապես հայտնի է, թե ինչ միջակայքում պետք է գտնվի արժեքը, դրանք ներկայացվում են Parameter Limits ֆայլերում։

# Տվյալների աբստրակտ տիպերը

  Ժամանակակից ծրագրային համակարգերը  նախագծվում են  մոդուլային սկզբունքով: Մոդուլի  իրականացման  եղանակը սովորաբար  թաքնված է օգտագործողից, իսկ  համագործակցությունը  մոդուլի  հետ  կատարվում է ինտերֆեյսի` մոդուլին  փոխանցվող  և  մոդուլից  ստացվող  պարամետրերի  միջոցով  Ժամանակակից ծրագրային համակարգերը  նախագծվում են  մոդուլային սկզբունքով: Մոդուլի  իրականացման  եղանակը սովորաբար  թաքնված է օգտագործողից, իսկ  համագործակցությունը  մոդուլի  հետ  կատարվում է ինտերֆեյսի` մոդուլին  փոխանցվող  և  մոդուլից  ստացվող  պարամետրերի  միջոցով: Այս մոտեցումը համարվում է ֆ‎‎‎‎‎‎‎‎‎ունկցիոնալ աբստրակցիայի իրականացման եղանակ: Ծրագրային  համակարգերը նախագծելիս   առկա է  նաև  տվյալների  աբստրակցիան: Ծրագրորդին սովորաբար  հայտնի են միայն այն  գործողությունները, որոնք  կարելի  է  կատարել  այդ  տվյալների  նկատմամբ: Օգտվողին ոչ միշտ են հասանելի ինչպես տվյալների ներկայացման, այնպես էլ   գործողությունների  իրականացման  ձևերը: Տվյալների կազմակերպման այդ սկզբունքի հետ  կապված է տվյալների  աբստրակտ  տիպ  հասկացությունը (Abstract Data Type, ADT): ADT  ասելով  հասկանում  ենք  տվյալների  մաթեմատիկական  մոդել այն գործողությունների համախմբով, որոնք  կատարվում են մոդելում ընդգրկված տվյալների  նկատմամբ: Այս մոտեցումը համարվում է ֆ‎‎‎‎‎‎‎‎‎ունկցիոնալ աբստրակցիայի իրականացման եղանակ: Ծրագրային  համակարգերը նախագծելիս   առկա է  նաև  տվյալների  աբստրակցիան: Ծրագրորդին սովորաբար  հայտնի են միայն այն  գործողությունները, որոնք  կարելի  է  կատարել  այդ  տվյալների  նկատմամբ: Օգտվողին ոչ միշտ են հասանելի ինչպես տվյալների ներկայացման, այնպես էլ   գործողությունների  իրականացման  ձևերը: Տվյալների կազմակերպման այդ սկզբունքի հետ  կապված է տվյալների  աբստրակտ  տիպ  հասկացությունը (Abstract Data Type, ADT): ADT  ասելով  հասկանում  ենք  տվյալների  մաթեմատիկական  մոդել այն գործողությունների համախմբով, որոնք  կատարվում են մոդելում ընդգրկված տվյալների  նկատմամբ: Տվյալների ADT աբստրակտ տիպը դիտարկենք որպես մաթեմատիկական մոդել իր գործողությունների բազմությամբ, որոնք որոշված են այդ մոդելի մեջ: ADT-ն կոնկրետ լեզվում իրականացնելիս օգտագործվում է տվյալների կառուցվածքներ:

Ծրագրի նախագծման և մշակման համար կարևոր է տվյալների աբստրակցիան: Աբստրակցիան որոշում է տվյալների կառուցվածքը և տվյալների հետ կատարվող գործողությունները: Աբստրակտ տիպի տվյալները կազմում են օգտվողի կողմից սահմանվող տիպ, որի գործողությունները ցույց են տալիս, թե օգտվողն ինչպես կարող է աշխատել այդ տվյալների հետ:

ADT-ն նկարագրվում է որոշակի ֆորմատով, որը պարունակում է վերնագիր, տվյալների տիպի նկարագիր, գործողությունների ցուցակ:

Յուրաքանչյուր գործողության համար որոշվում են`

* Մուտքային արժեքներ (input )
* Նախապայմաններ (preconditions), որոնց բավարարում են տվյալները նախքան նրանց հետ գործողություն կատարելը,
* Պրոցես (process), որը նկարագրում է կատարվող գործողությունը,
* Ելքային արժեքներ (output),
* Վերջնապայման (postconditions), որոնց բավարարում են տվյալները գործողությունը կատարելուց հետո:

ADT-ների մեծ մասն ունեն սկզբնարժեքավորման գործողություններ (initializer), որոնք տվյալներին տալիս են սկզբնական արժեքներ: C++ լեզվում տվյալների սկզբնարժեքավորումը իրականացվում է դասի կոնստրուկտորի (constructor) միջոցով:

1.3 Տվյալների հետ կատարվող գործողությունները

Որոնում

Որոնումը կոմպյուտերային ծրագրավորման ամենից շատ հանդիպող գործողություններից է։ Այն նաև իդեալական խնդիր է զանազան տվյալների կառուցվածքները փորձարկելու համար։ Գոյություն ունեն որոնման թեմայի մի քանի վարիացիաներ, և մշակված են բազմաթիվ ալգորիթմներ։ Հետևյալ քննարկումներում ենթադրվում է, որ տվյալների հավաքածուն, որի մեջ որոնվելու է տրված տարրը, ֆիքսված է։ Կհամարենք, որ N տարր պարունակող բազմությունը տրված է, օրինակ, զանգվածի տեսքով․

a : ARRAY N OF Item

Սովորաբար Item տիպը նկարագրված է որպես գրառում (record), որի դաշտերից մեկը բանալին (key) է։ Խնդիրը բերվում է այն տարրի գտնելուն, որի բանալին հավասար է x որոնման արգումենտին։ Արդյունքում ստացված iինդեքսը, որ բավարարում է a[i].key = x պայմանին, հնարավորություն է տալիս դիմելու գտնված տարրի մյուս դաշտերին։ Քանի որ մեզ հետաքրքրում է միայն որոնման խնդիրը, և առայժմ չենք մտածում այն մյուս տվյալների մասին, որոնց համար որոնվում էր տարրը, կենթադրենք, որ Item տիպը պարունակում է միայն բանալին, այսինքն այն ինքը հենց բանալին է։

Գծային որոնում (linear search)

Եթե որոնվող տվյալների մասին այլ լրացուցիչ տեղեկություններ տրված չեն, ապա ակնհայտ մոտեցումը զանգվածի տարրերի հաջորդական դիտարկումն է, քայլ առ քայլ մեծացնելով նրա այն հատվածի չափը, որտեղ որոնվող տարրը հայտնաբերված չէ։ Այս մոտեցումը կոչվում է գծային որոնում։ Որոնման ավարտի պայմանները երկուսն են․

Տարրը գտնված է, այնսինքն՝ a[i] = x։

Դիտարկված է ամբողջ զանգվածը և համընկնում չի հայտնաբերված։

Ստացվում է հետևյալ ալգորիթմը․

i := 0;

WHILE (i < N) & (a[i] # x) DO INC(i) END

ՈՒշադրություն դարձրեք, որ բուլյան արտահայտության մեջ ենթաարտահայտությունների կարգը կարևոր է։

Ցիկլի ինվարիանտը, այսինքն այն պայմանը, որը ճշմարիտ է ցիկլի ամեն մի իտերացիայի սկզբում և վերջում, այսպիսինն է․

(0 ≤ i < N) & (A k: 0 ≤ k < i: a[k] ≠ x)

Այն ցույց է տալիս, որ k֊ից փոքր բոլոր i֊երի համար համընկնումներ չեն եղել։ Ուշադրություն դարձնենք նաև, որ ցիկլի ամեն մի իտերացիայից առաջ և հետո i֊ի արժեքները տարբեր են։ Այնուամենայնիվ, ինվարիանտը պահպանվում է ցիկլի պայմանում։

Այս ասվածից և այն փաստից, որ որոնումն ավարտվում է միայն այն ժամանակ, երբ ցիկլի պայմանը տեղի չունի (կեղծ է), կարելի է դուրս բերել վերջնական պայմանը․

((i = N) OR (a[i] = x)) & (A k: 0 ≤ k < i: a[k] ≠ x)

Այս պայմանը ոչ միայն մեր ցանկալի արդյունքն է, այլ դրանից նաև հետևում է, որ եթե ալգորիթմը գտել է համընկնում, ապա այն գտել է ամենափոքր ինդեքսովը, այսինքն՝ առաջինը որոնվող տարրերից։ i = N հավասարությունը ցույց է տալիս, որ համընկնումներ չեն հայտնաբերվել։

Ցիկլի կրկնությունների ավարտը ակնհայտորեն երաշխավորված է, որովհետև ամեն մի քայլում i֊ի արժեքն աճում է, հետևաբար այն, իհարկե, վերջավոր քայլերից հետո կհասնի N֊ին։ Իրականում, եթե համընկնումներ չեն եղել, ապա դա տեղի կունենա N քայլից հետո։

Պարզ է, որ յուրաքանչյուր քայլում պահանջվում է մեծացնել ինդեքսը և հաշվել բուլյան արտահայտությունը։ Կարելի՞ է արդյոք այս խնդիրը պարզեցնել, և դրանով պարզեցնել նաև որոնման գործողությունը։ Միակ հնարավորությունը երկու կտորից բաղկացած բուլյան արտահայտություը պարզեցնելու մեջ է։ Հետևաբար, միակ տարբերակը այնպիսի ավելի պարզ պայմանի ձևակերպումն է, որը համարժեք է մեր ունեցած բարդ պայմանին։ Դա հնարավոր է, եթե երաշխավորենք, որ համընկնում միշտ տեղի է ունենալու, որին կարելի է հասնել զանգվածի վերջում x արժեքով լրացուցիչ տարր ավելացնելով։ Այս լրացուցիչ տարրը կանվանենք պատնեշ (sentinel), որովհետև այն ինդեքսին արգելում է դուրս ելնել զանգվածի սահմաններից։ Այժմ a զանգվածը սահմանված է որպես․

a: ARRAY N+1 OF INTEGER

իսկ գծային որոնման ալգորիթմը, պատնեշի օգտագործմամբ, կունենա հետևյալ տեսքը․

a[N] := x; i := 0;

WHILE a[i] # x DO INC(i) END

Նույն ինվարիանտից դուրս բերված վերջնական պայմանը կլինի․

(a[i] = x) & (A k: 0 ≤ k < i: a[k] ≠ x)

Պարզ է, որ i = N պայմանը ցույց է տալիս, որ համընկնումներ չեն եղել՝ բացառությամբ պատնեշի հետ համընկնելը։

Որոնում կիսման ողանակով (binary search)

Լրիվ ակնհայտ է, որ այլևս հնարավորություն չկա արագացնելու որոնման գործողությունը, եթե միայն որոնվող տարրերի մասին տրված չեն լրացուցիչ տեղեկություններ։ Հայտնի է, որ որոնումը կարելի է ավելի արդյունավետ դարձնել, եթե տվյալները կարգավորված են։ Պատկերացրեք, օրինակ, մի հեռախոսագիրք, որում անունները այբբենական կարգով դասավորված չեն։ Դա լրիվ անպետք բան է։ Մենք կներկայացնենք ալգորիթմ՝ հիմնված a զանգվածում տարրերի կարգավորված լինելու փաստի վրա, այսինքն՝ հաշվի առնելով հետևյալ պայմանը․

A k: 1 ≤ k < N: a[k-1] ≤ a[k]

Հիմնական գաղափարն է՝ վերցնել մի պատահական տարր, օրինակ, a[m] և այն համեմատել x որոնման արգումենտի հետ։ Եթե այն հավասար է x֊ին, ապա որոնումն ավարտվում է, եթե այն փոքր է x֊ից, ապա m֊ից փոքր կամ հավասար ինդեքս ունեցող բոլոր տարրերը կարելի է բացառել հետագա որոնումից, և եթե այն մեծ է x֊ից, ապա կարելի է բացառել m֊ից մեծ և հավասար ինդեքս ունեցող տարրերը։ Ասվածից հանգում ենք կիսման եղանակով որոնման

ալգորիթմին։ Այն օգտագործում է L և R ինդեքսային փոփոխականները, որոնք ցույց են տալիս զանգվածի այն հատվածի ձախ և աջ սահմանները, որոնցում դեռ կարող է հայտնաբերվել որոնվող տարրը։

L := 0; R := N - 1;

m := Lի և Rի միջև ընկած որևէ տարր;

WHILE (L <= R) & (a[m] # x) DO

IF a[m] < x THEN

L := m + 1

ELSE

R := m - 1

END;

m := Lի և Rի միջև ընկած որևէ տարր

END

Ուշադրություն դարձրեք այս ալգորիթմի և նախորդ բաժնում նկարագրված գծային որոնման ալգորիթմի հիմնական կառուցվածքային նմանությանը․ այստեղ i ինդեքսի դերը կատարում է L, m, R եռյակը։ Այդ նմանությունը բացատրելու, և, այնուհետև, ցիկլի կոռեկտության մեջ ավելի լավ համոզվելու համար, մենք ձեռնպահ մնացինք փոքրիկ օպտիմիզացիայից, որը կբացառեր mի երկու նույնական վերագրումները։

Ցիկլի ինվարիանտը, այսինքն՝ ամեն մի քայլից առաջ ու հետո տեղի ունեցող պայմանը, հետևյալն է․

(A k: 0 ≤ k < L: a[k] < x) & (A k: R < k < N: a[k] > x)

որից դուրս է բերված հետևյալ արդյունքը․

((L > R) OR (a[m] = x)) & (A k: 0 ≤ k < L: a[k] < x) & (A k: R < k < N: a[k] > x)

որից էլ հետևում է․

((L > R) & (A k: 0 ≤ k < N: a[k] ≠ x)) OR (a[m] = x)

m ինդեքսի ընտրությունը բոլորովին կամայական է այն իմաստով, որ ալգորիթմի կոռեկտությունը դրանից կախված չէ։ Բայց m֊ի ընտրությունն ազդում է ալգորիթմի արդյունավետության վրա։ Պարզ է, որ մեր նպատակն է ցիկլի ամեն մի քայլում հետագա որոնումից բացառել որքան հնարավոր է շատ տարրեր՝ անկախ համեմատման արդյունքից։ Լավագույն լուծումը կենտրոնական տարրի ընտրությունն է, որովհետև ամեն մի քայլում այն բացառում է զանգվածի տարրերի կեսը։ Արդյունքում քայլերի առավելագույն քանակը հավասար է log2 N, կլորացված վերև՝ մինչև ամենամոտիկ ամբողջ թիվը։ Այսպիսով, այս ալգորիթմը էականորեն շահեկան է գծային որոնման ալգորիթմից, որում համեմատությունների սպասվող քանակը N/2 է։

Արդյունավետությունը կարելի է մի քիչ լավացնել, տեղերով փոխելով ցիկլի մարմնի IF պայմանները։ Հավասարությունը պետք է ստուգել երկրորդ հերթին, որովհետև այն հանդիպում է միայն մեկ անգամ և բերում է ցիկլի ավարտին։ Բայց ավելի կարևոր է այն հարցը, թե արդյոք, ինչպես գծային որոնման դեպքում, հնարավո՞ր է գտնել ցիկլի ավարտը որոշող ավելի պարզ պայման։ Իսկապես մենք գտնում ենք այդպիսի արագ ալգորիթմ, հենց որ հրաժարվում ենք որոնման ցիկլը տարրերի համընկնումով ավարտելու միամիտ ցանկությունից։ Առաջին հայացքից սա տարօրինակ է թվում, բայց ավելի ուշադիր ուսումնասիրությամբ բացահայտվում է, որ ամեն մի քայլում արդյունավետության շահն ավելին է, քան մի քանի լրացուցիչ տարրերի համեմատումից ստացված կորուստները։ Հիշենք, որ քայլերի առավելագույն քանակը log N է։

Արագ լուծումը հիմնված է հետևյալ ինվարիանտի վրա․

(A k: 0 ≤ k < L: a[k] < x) & (A k: R ≤ k < N: a[k] ≥ x)  
և որոնումը շարունակվում է այնքան ժամանակ, քանի դեռ երկու հատվածները չեն ծածկել ամբողջ զանգվածը։

L := 0; R := 0;

WHILE L < R DO

m := (L + R) DIV 2;

IF a[m] < x THEN L := m + 1 ELSE R := m END

END

Ավարտի պայմանն է L ≥ R։ Բայց հասանելի՞ է արդյոք այն։ Այս պնդումն ապացուցելու համար պետք է ցույց տանք, որ բոլոր դեպքերում R - L տարբերությունը նվազում է ամեն մի քայլից հետո։ L < R տեղի ունի յուրաքանչյուր քայլի սկզբում։ m թվաբանական միջինի կամար ճշմարիտ է L ≤ m < R։ Հետևաբար, տարբերությունն իրոք նվազում է L֊ին m+1 վերգրելով (Լ-ը մեծացնելով), կամ R֊ին m վերագրելով (R֊ը նվազեցնելով), և ցիկլի կրկնությունն ավարտվում է L = R պայմանով։

Այնուամենայնիվ, մեր ինվարիանտը և L = R պայմանը դեռևս համընկնում չեն նշանակում։ Իհարկե, եթե R = N, ապա համընկնում չկա։ Այլապես պետք է հաշվի առնենք, որ a[R] տարրը երբեք չի համեմատվել։ Հետևաբար, մի լրացուցիչ a[R] = x հավասարության ստուգումն անհրաժեշտ է։ Ի տարբերություն առաջին լուծման, այս ալգորիթմը, ինչպես և գծային որոնումը, գտնում է ամենափոքր ինդեքսով համընկնող տարրը։

Կարգավորում (sorting)

Կարգավորում ասելով սովորաբար հասկացվում է օբյեկտների տրված բազմությունը ըստ որևէ կարգի վերադասավորելու գործողությունը։ Կարգավորման նպատակը կարգավորված բազմության մեջ տարրի հետագա որոնման հեշտացումն է։ Որպես այդպիսին այն համարյա համապիտանի, ֆունդամենտալ գործողություն է: Հետևաբար, կարգավորումը անհրաժեշտ և կարևոր գործողություն է, հատկապես տվյալները մշակելու համար։ Այնուամենայնիվ, կարգավորման նկատմամբ մեր առաջնային հետաքրքրություն հիմնված է այն բանին, որ ալգորիթմները կազմելիս շփվում ենք բազմաթիվ հիմնարար գաղափարների հետ։ Շատ չեն այն հնարքները, որոնք չեն հանդիպում կարգավորման ալգորիթմների հետ կապված խնդիրներում։ Մասնավորապես, կարգավորումը դա ալգորիթմների բազմազանության ցուցադրման իդեալական միջոց է։ Բոլոր այդ ալգորիթմները ծառայում են նույն նպատակին, դրանցից որոշները ինչ-որ իմաստով արդյունավետ են, և դրանցից շատերը առավելություններ ունեն մյուսների նկատմամբ։ Կարգավորումը նաև իդեալական միջոց է՝ ցույց տալու ալգորիթմների արդյունավետության վերլուծության կարևորությունը։ Շատ հարմար է նաև կարգավորման օրինակով ցույց տալ, թե ինչպես կարելի է արտադրողականության մեջ զգալիորեն շահել՝ մշակելով ավելի բարդ ալգորիթմներ, թեև ձեռքի տակ արդեն ունենալով արդեն ակնհայտ մեթոդներ։ Ալգորիթմի ընտրությունը կախված է մշակվող տվյալների առանձնահատկություններից՝ սա համարյա օրենք է, սակայն կարդավորման դեպքում այդ կախվածությունն այնքան խորն է, որ կարգավորման մեթոդները բաժանվում են երկու դասի՝ զանգվածների կարգավորում և (հաջորդական) ֆայլերի կարգավորում։ Հաճակ այս երկու դասերն անվանվում են ներքին և արտաքին կարգավորում, քանի որ զանգվածները պահվում են համակարգչի արագագործ, կամայական-դիմման «ներքին» հիշասարքում(RAM), իսկ ֆայլերը պահվում են ավելի դանդաղագործ, բայց ավելի տարողունակ «արտաքին» հիշասարքերում(Hard Drive,SSD)՝ կառուցված մեխանիկական շարժվող մասերով (սկավառակ, ժապավեն)։   
Որոշ սահմանումներ ու հասկացություններ ներմուծենք, որոնք պետք է օգտագործվեն այս գլխի շարադրանքում։ Եթե տրված են n տարրեր՝

a0, a1, ... , an-1

ապա կարգավորել, նշանակում է այդ տարրերը դասավորել

ak0, ak1, ... , ak[n-1]

զանգվածում այնպես, որ տրված f կարգավորող ֆունկցիայի համար տեղի ունեն հետևյալ անհավասարությունները՝

f(ak0) ≤ f(ak1) ≤ ... ≤ f(ak[n-1])

Զանգվածների կարգավորման մեթոդների ներկայացվող հիմնական պահանջը մատչելի հիշողության խնայողական օգտագործումն է։ Սա ենթադրում է, որ տարրերը կարգավորվածության բերելու տեղափոխությունները պետք է կատարվեն նույն տեղում (in place), այսինքն՝ a զանգվածից b նպատակային զանգվածը տարրերը տեղափոխող մեթոդները ներկայացնում են նվազագույն հետաքրքրություն։ Բազմաթիվ հնարավոր լուծումների մեջ մեզ հարկավոր եղածի ընտրությունը սահմանափակելով հիշողության խնայողության պահանջով՝ մեթոդները նախ կդասակարգենք ըստ իրեն խնայողականության, այսինքն՝ ըստ աշխատելու ժամանակի։ Չնայած որ կարգավորման լավ մեթոդները պահանջում են մոտավորապես n\*log n համեմատություններ, մենք նախ կքննարկենք մի քանի պարզ ու ակնհայտ մեթոդներ, դրանց անվանում են նաև ուղիղ մեթոդներ, որոնցում պահանջվում են բանալիների մոտ n^2 համեմատություններ։ Երեք կարևոր պատճառներ կան, որոնք ստիպում են այս մեթոդներն ուսումնասիրել ավելի արագ մեթոդներից առաջ։

* Ուղիղ մեթոդները հատկապես հարմար են կարգավորումների մեծամասնության հիմնական սկզբունքների բնութագրական գծերը բացատրելու համար։
* Այդ մեթոդների ծրագրերը հեշտ հասկանալի և կարճ են։ Հիշեցնենք, որ ծրագրերը նույնպես հիշողություն են զբաղեցնում։
* Բարդ մեթոդները պահանջում են ոչ մեծ քանակի գործողություններ, որոնցից ամեն մեկը նույնպես բավականին բարդ է, և այդ պատճառով էլ բավականաչափ փոքր n-երի համար պարզվում է, որ ուղիղ մեթոդները կարող են ավելի արագ լինել, թեև մեծ n-երի համար չարժե ուղիղ մեթոդ օգտագործել։

Տարրերը տեղում կարգավորաղ մեթոդներից ամենահայտնիներն են՝

* Տեղադրմամբ կարգավորում (insertion sort)
* Ընտրությամբ կարգավորում (selection sort)
* Պղպջակներով կարգավորում (bubble sort)

Բանալիների բաշխում (Հեշավորում - Hashing)

Սկզբունքային հարցը հետևյալն է․եթե տրված է բանալիով (key) բնութագրվող տարրերի բազմություն , ապա ինչպե՞ս կազմակերպել այդ բազմությունը, որպեսզի տրված բանալիով տարրին դիպելու համար պահանջվի նվազագույն ջանք։ Պարզ է, որ համակարգչի հիշողության մեջ գտնվող որևէ տարրին դիմելու համար պետք է ունենալ նրա հասցեն։ Հետևաբար, նշված խնդիրը բերվում է K բանալիները A հասցեներին արտապատկերող H ֆունկցիայի սահմանմանը։

H: K → A

Ներկայացնենք ըստ էութան պարզ և շատ դեպքերում արդյունավետ մի եղանակ։ Այնուհետև կքննարկենք նաև այս եղանակի որոշ թերություններ։ Այս մոտցման մեջ տվյալները կազմակերպվում են զանգվածի տեսքով։ Այս դեպքում H-ը բանալիները զանգվածի ինդեքսներին բաշխող արտապատկերում է, որտեղից էլ հենց առաջացել է այս մեթոդի համար օգտագործվող բանալիների բաշխում անվանումը։ Պետք է նշել, որ այս դեպքում կարիք չունենք հիշողության դինամիկ առանձնացնող գործողությունների․զանգվածը միակն է և կազմված է ստատիկ կառուցվածքներից։ Բանալիների բաշխման մեթոդը հաճախ օգտագործվում է այն խնդիրներում, որտեղ մոտավորապես նույն հաջողությամբ կարելի է օգտագործել նաև ծառաձև կառուցվածքները։

Բանալիների բաշխման եղանակն օգտագործելիս հիմնական դժվարությունն այն է, որ թույլատրելի բանալիների բազմությունն էապես ավելի մեծ է, քան հիշողության մատչելի հասցեները (զանգվածի ինդեքսները)։ Որպես բանալի վերցնենք, օրինակ, մինչև 16 նիշ պարունակող անունները, որոնք հազարավոր մարդկանց մեջ նույնականացնում են առանձին անհատների։ Այսինքն, գոյություն ունեն 26^16 հնարավոր բանալիներ, որոնք պետք է արտապատկերել 10^3 հնարավոր ինդեքսների։ Ակնհայտ է, որ այս դեպքում H-ը շատը-մեկին ֆունկցիա է։ Եթե տրված է k բանալին, ապա որոնման գործողության առաջին քայլը h=H(k) ինդեքսի հաշվարկումն է, իսկ երկրորդ, ակնհայտորեն պարտադիր, քայլը ստուգելն է, թե արդյո՞ք k բանալիով տարրը համապատասխանում է T զանգվածի (աղյուսակի) h ինդեքսով տարրին, այսինքն, ստուգել T[H(k)].key = k հավասարությունը։ Անմիջապես հանդիպում ենք հետևյալ երկու հարցերին․

* Ինչպիսի՞ն պետք է լինի H ֆունկցիան։
* Ի՞նչ անել, եթե H ֆունկցիան չի կարողացել հաշվել որոնելի տարրի դիրքը։

Երկրորդ հարցի պատասխանը կայանում է նրանում, որ պետք է ունենալ մի մեթոդ, որը վերադարձնում է այլընտրանաքային դիրք, ասենք h', եթե այդ նոր դիրքն էլ նորից չի պարունակում որոնելի տարրը, ապա վերադարձնում է h'' դիրքը, և այդպես շարունակ։ Եթե հաշվարկված դիրքում գտնվում է որոնելի տարրից տարբերվող մի այլ տարր, ապա այդ իրավիճակը կոչվում է ընդհարում (collision), իսկ այլընտրանքային դիրքի հաշվարկը՝ ընդհարման լուծում։ Ստորև կքննարկենք բանալիների բաշխման ֆունկցիայի ընտրությունը և ընդհարման լուծման եղանակները։

Հեշավորող ֆունկցիայի ընտրությունը

Լավ բաշխման ֆունկցիայի համար կարևոր նախապայմանն է, որ այն կարողանա բանալիների բազմությունը հնարավորինս հավասարաչափ բաշխել ինդեքսների արժեքների ամբողջ միջակայքին։ Այս պայմանից բացի բաշխման համար այլ սահմանափակումներ չկա, սակայն իրականում ցանկալի է, որ այն նման լինի պատահական բաշխման։ Այսպիսի առանձնահատկությունը բանալիների բաշխման մեթոդին տվել է թերևս ոչ-գիտական մի անվանում՝ հեշավորում (hashing), այսինքն՝ արգումենտի կտրտում, H-ն էլ կոչվում է հեշավորող ֆունկցիա (hash function)։ Այս ֆունկցիան պետք է լինի արդյունավետ հաշվարկվող, այսինքն՝ բաղկացած լինի շատ քիչ թվով թվաբանական գործողություններից։ Դիցուք կարող ենք օգտագործել ORD(k) ֆունկցիան, որը վերադարձնում է k բանալու հերթական համարը՝ կարգաթիվը, բոլոր հնարավոր բանալիների բազմության մեջ։ Ենթադրենք նաև, որ զանգվածի i ինդեքսը տարածվում է 0-ից մինչև N-1, որտեղ N-ը զանգվածի տարրերի քանակն է։ Այս դեպքում ընտրությունն ակնհայտ է՝

H(k) = ORD(k) MOD N

Այս ֆունկցիան ապահովում է բանալիների հավասարաչափ բաշխումն ինդեքսների ամբողջ միջակայքում և հենց դրա համար էլ օգտագործվում է շատ հեշավորող ֆունկցիաներում։ Այն նաև շատ արդյունավետ հաշվարկվում է, եթե N-ը երկուսի աստիճան է։ Սակայն եթե բանալին տառերի հաջորդականություն է, ապա հենց այսպիսի ֆունկցիայից պետք է խուսափել։ Բանն այն է, որ այս դեպքում բոլոր բանալիների՝ հույն հավանականությամբ հանդիպելու ենթադրությունը սխալ է։ Իրականում բառերը, որոնք տարբերվում են միայն մի քանի տառերով, մեծ հավանականությամբ կարող են արտապատկերվել նույն ինդեքսին, որն էլ կստեղծի խիստ անհավասարաչափ բաշխում։ Այս պատճառով էլ գործնականում խորհուրդ է տրվում N-ը վերցնել մի որևէ պարզ թիվ ։ Որպես հետևանք արդեն հարկ է լինում օգտագործել լրիվ բաժանման գործողություն, որը հնարավոր չէ փոխարինել երկուական թվանշանների կրճատումով։ Սակայն ժամանակակից համակարգիչներում սա դժվար խնդիր չէ, քանի որ դրանցում առկա է ներդրված բաժանման գործողություն։

Հաճախ օգտագործվում են բանալու որևէ հատվածի երկուական ներկայացման վրա կիրառվող տրամաբանական գործողություններից կառուցված հեշավորող ֆունկցիաներ, օրինակ` «բացառող կամ»(XOR)։ Որոշ համակարգիչների վրա այս գործողությունները կարող են կատարվել ավելի արագ, քան բաժանումը, բայց հաճախ դրանք բերում են ինդեքսերի միջակայքում բանալիների զարմանալի անհավասարաչափ բաշխման։ Այդ պատճառով էլ ձեռնպահ կմնանք այդպիսի մեթոդների հետագա քննարկումից։

Բախումների լուծումը

Եթե պարզվում է, որ աղյուսակի՝ տրված բանալուն համապատասխանեցված տարրը որոնելին չէ, ապա տեղի ունի բախում, այսինքն՝ երկու տարբեր տարրերի բանալիներ արտապատկերվում են նույն ինդեքսին։ Այս դեպքում անհրաժեշտ է երկրորդ փորձը՝ տրված բանալուց խիստ որոշակի եղանակով ստացվող ինդեքսի օգտագործմամբ։ Գոյություն ունեն երկրորդային ինդեքսների հաշվարկման մի քանի մեթոդներ։ Ակնհայտ եղանակ է՝ կապակցված ցուցակում հավաքել նույն H(k) առաջնային ինդեքս ունեցող տարրերը։ Սա կոչվում է ուղիղ կապակցում (direct chaining)։ Այս ցուցակի տարրերը կարող են առաջնային աղյուսակում լինել կամ չլինել, երկրորդ դեպքում դրանց զբաղեցրած հիշողության տիրույթը կոչվում է գերբեռնվածության տիրույթ (overflow area)։ Այս հնարքի թերությունն այն է, որ պետք է կազմակերպել երկրորդային ցուցակների հետ աշխատանքը, ինչպես նաև աղյուսակի ամեն մի տարրը պետք է ցուցիչ (կամ ինդեքս) ունենա խնդրահարույց տարրերի ցուցակի համար։ Բախումների լուծման այլընտրանքային եղանակ է՝ ընդհանրապես հրաժարվել ցուցակներից և պարզապես դիտարկել նույն աղյուսակի մյուս տարրերը, մինչև կգտնվի տարրը կամ կգտնվի ազատ դիրք։ Վերջին դեպքում համարում ենք, որ որոնվող տարրն աղյուսակում բացակայում է։ Այս եղանակը կոչվում է բաց հասցեավորում (open addressing)։ Բնականաբար, տրված բանալու համար երկրորդային փորձերի ինդեքսների հաջորդականությունը միշտ պետք է նույնը լինի։ Ըստ ասվածի, որոնման ալգորիթմը կարող է ունենալ հետևյալ տեսքը․

h := H(k); i := 0;

REPEAT

IF T[h].key = k THEN տարր գտնված է

ELSIF T[h].key = free THEN տարրն աղյուսակում չէ

ELSE (\* բախում \*)

i := i+1; h := H(k) + G(i)

END

UNTIL գտնված կամ աղյուսակում չէ (կամ աղյուսակը լիքն է)

Բախումների լուծման տարբեր եղանակներ են առաջարկվել գրականության մեջ։ Մորիսի (Morris) կողմից 1968-ին թեմայի ուսումնասիրությունը բավականին աշխուժություն բերեց ոլորտ։ Ամենապարզ ձևը՝ դիտարկել աղյուսակի հերթական տարրը (համարենք աղյուսակը շրջանաձև), մինչև կգտնվի տրված բանալիով տարրը կամ կգտնվի դատարկ տեղ։ Այսպիսով, G(i) = i, իսկ հաջորդկան փորձերի համար օգտագործվող h\_i ինդեքսները որոշվում են հետևյալ կերպ․

h\_0 = H(k)

h\_i = (h\_i-1 + i) MOD N, i = 1 ... N-1

Սա կոչվում է գծային փորձերի եղանակ (linear probing)։ Նրա թերությունն այն է, որ տարրերը ձգտում են կուտակվել առաջնային բանալիների շուրջը (այսինքն՝ այն բանալիների շուրջը, որոնք ավելացնելիս բախում չի առաջացել)։ Իհարկե, իդեալական դեպքում G ֆունկցիան էլ պետք է բանալիները հավասարաչափ բաշխի ազատ դիրքերի բազմության վրա։ Սակայն գործնականում բավականին դժվար է ապահովել այդ պահանջը, և այս դեպքում նախընտրում են փոխզիջումային մեթոդներ, որոնք պահանջում են բարդ հաշվարկներ, բայց ավելի լավն են քան գծային ֆունկցիան։ Դրանցից մեկում օգտագործվում է քառակուսային ֆունկցիան այնպես, որ հաջորդական փորձերի ինդեքսները որոշվում են հետևյալ կերպ․

h\_0 = H(k)

h\_i = (h\_0-1 + i^2) MOD N, i > 0

Նկատենք, որ հերթական ինդեքսը հաշվելիս կարելի է ազատվել քառակուսի բարձրացնելու գործողությունից, եթե h\_i = i^2 և d\_i = 2i + 1 համար օգտագործենք հետևյալ անդրադարձ առնչությունները՝

h\_i+1 = h\_i + d\_i

d\_i+1 = d\_i + 2, i >0

որտեղ h\_0 = 0 և d\_0 = 1։ Սա կոչվում է քառակուսային փորձերի մեթոդ (quadratic probing), և ընդհանուր դեպքում այն շրջանցում է վերը նշված կուտակումների խնդիրը՝ գործնականում չպահանջելով լրացուցիչ հաշվարկներ։ Մի փոքր թերությունն այն է, որ հաջորդական փորձերի ժամանակ աղյուսակի ոչ բոլոր տարրերն են ստուգվում, այսինքն տարրն ավելացնելիս հնարավոր է չնկատել ազատ դիրքը, թեև աղյուսակում այդպիսիք կան։ Իրականում, եթե N-ը պարզ թիվ է, ապա քառակուսային փորձերի մեթոդով ստուգում է աղյուսակի ամենաքիչը կեսը։ Այս պնդումը կարելի է դուրս բերել հետևյալ կերպ։ Այն փաստը, որ i-րդ և j-րդ փորձերը բերում են այսուսակի նույն տարրին, արտահայտվում է հետևյալ հավասարությամբ․

i^2 MOD N = j^2 MOD N

(i^2 - j^2) ≡ 0

Քառակուսիների տարբերությունն արտահայտելով երկու արտադրիչներով ստանում ենք`

(i + j)(i - j) ≡ 0

և քանի որ i ≠ j, ապա եզրակացնում ենք, որ i և j թվերից գոնե մեկը պետք է փոքր լինի N/2-ից, որպեսզի ամբողջաթիվ c-ի համար ստանանք i + j = c \* N։ Գործականում այս թերությունն էական չէ, քանի որ բախումների լուծման համար N/2 փորձեր հազվադեպ են կատարվում, այն էլ միայն այն դեպքում, երբ աղյուսակը համարյա լիքն է։ Բանալիների բաշխման մեթոդի վատագույն դեպքում բանալին ավելացնելու և որոնելու արտադրողականությունը սարսափելի է։ Լրիվ հնարավոր է, որ որոնման արգումենտնը փորձերի ժամանակ անցնի բոլոր զբաղված դիրքերով՝ ոչ մի անգամ չընկնելով աղյուսակի անհրաժեշտ (կամ ազատ) դիրքի վրա։ Հեշավորման տեխնիկան օգտագործելու համար պետք է բավականաչափ վստահ լինել հավանականության տեսության օրենքներին։ Մենք պարզապես ուզում ենք համոզված լինել, որ փորձերի միջին թիվը փոքր է։ Ստորև բերված հավանակային փաստարկներն ցույց են տալիս, որ այդ թիվը ոչ թե փոքր է, այլ շատ փոքր է։  
Մեկ անգամ էլ ենթադրենք, որ բոլոր բանալիների հանդիպելու հավանականությունը հավասար է, իսկ H հեշ-ֆունկցիան դրանք հավասարաչափ բաշխում է աղյուսակի ինդեքսների միջակայքի վրա։ Ենթադրենք նաև, թե բանալին պետք է ավելացվի արդեն k հատ տարրեր պարունակող և N չափ ունեցող աղյուսակի մեջ։ Այդ դեպքում առաջին փորձից ազատ դիրքի վրա ընկնելու հավանականությունը (N-k)/N է։ Սա նաև միայն մեկ համեմատում կատարելու p\_1 հավանականությունն է։ Հավանականությունը, որ պետք կլինի ճիշտ մեկ երկրորդային փորձ, հավասար է առաձին փորձում բախման հավանականությանը՝ բազմապատկած երկրորդ փորձում ազատ դիրքի վրա ընկնելու հավանականությանը։ Այսպիսով, ստանում ենք ճիշտ i փորձեր կատարելու p\_i հավանականության հաշվման հետևյալ եղանակը․

p\_1 = (N-k)/N

p\_2 = (k/N) × (N-k)/(N-1)

p\_3 = (k/N) × (k-1)/(N-1) × (N-k)/(N-2)

..........

p\_i = (k/N) × (k-1)/(N-1) × (k-2)/(N-2) × ... × (N-k)/(N-(i-1))

Այստեղից էլ աղյուսակում k+1-րդ բանալին ավելացնելու փորձերի E քանակը հավասար է

E\_k+1 = \_\_S\_\_i: 1 ≤ i ≤ k+1 : i × pi = 1 × (N-k)/N + 2 × (k/N) × (N-k)/(N-1) + ... + (k+1) \* (k/N) × (k-1)/(N-1) × (k-2)/(N-2) × ... × 1/(N-(k-1)) = (N+1) / (N-(k-1))

# 1.4 QT գրադարանի տարրերը

Qt-ն C++ լեզվի հիման վրա կառուցված անկախ համակարգչի պլատֆորմից և օպերացիոն համակարգից գրադարան է: Այն կառուցված է հետևյալ մոտեցումամբ՝ բոլոր դասերը ժառանգվում են մեկ QObject դասից: Այն ծրագրավորողին լայն հնարավորություն է տալիս գրաֆիկական միջավայր ստեղծելու համար: Այսօր Qt-ն բավականին ընդգրկուն է դարձել և արդեն հնարավորություն ունի աշխատելու ոչ միայն գրաֆիկական տարբեր օբյեկտների հետ, այլ նաև տվյալների բազաների, սերվերների, ինտերնետային տարբեր տեխնոլոգիաների հետ: Այն նաև ունի իր կոնտեյներները, որոնք բավականին նման են STL գրադարանի կոնտեյներներին: QT-ն հիմնված է մի քանի հիմնաքարերի վրա՝

* MetaObject մեխանիզմ: Այս մեխանիզմի միջոցով կարող ենք դինամիկ կերպով ինֆորմացիա ստանալ օբյեկտի մասին:
* QObject դասը, լինելով ժառանգման հիերարխիայի ամենավերևում ապահովում է meta-object մեխանիզմի առկայությունը անհրաժեշտ դեպքերում
* Q\_OBJECT macro-ն, որը գրված է լինում դաս-ի փակ դաշտում meta-object մեխանիզմն է ակտիվացնում:
* Meta-Object compiler-ը անհրաժեշտության դեպքում QObject դասի ժառանգներին ձևափոխում է moc օբյեկտների, որոնցում մետաինֆորմացիան ներկայացվում է զուտ C++ լեզվով:

Համապատասխան կառուցվածքները իրագործելու համար օգտագործվել են QT-ի կոնտեյներներից QMap, QHash, QVector, QString, QList, QVariantList դասերը։

Գլուխ 2  
Ծրագրային իրագործում

2.1 Տվյալների պահպանման ինտերֆեյսները

Ծրագրի կատարման ընթացքում ֆայլերը վերլուծություն անցնելուց հետո պետք է պահպանվեն հիշողության մեջ՝ մշակված կառուցվածքների միջոցով։ Տվյալների հիմնական քեշը, որը ամենագլխավորն է իր նշանակությամբ, DataStore դասն է, որին կանդրադառնանք ավելի ուշ։

Բոլոր տվյալները, որոնք վերլուծության (parsing) ընթացքում ստացվել են, իրենցից ներկայացնում են սյունակներ, որոնց էլ ծրագիրը պահելու է IVector ինտերֆեյսի և նրա բազում իրականացումների (implementation) մեջ։ Դրանք են՝

|  |  |
| --- | --- |
| **Interface** | **Implementation** |
| IIntVector | CIntData, CIntDataGroup |
| IDoubleVector | CDoubleVector, CDoubleVectorGroup |
| IBoolVector | CBoolData, CBoolDataGroup |
| IStringVector | CStringData, CStringDataGroup |
| IDateTimeVector | CDateTimeData, CDateTimeDataGroup |

Վերը թվարկված բոլոր դասերը հանդիսանում են IVector ինտերֆեյսի ժառանգ դասեր։

IVector-ը իրենից ներկայացնում է հետևյալ դասը՝

class IVector

{

public:

virtual QVariant GetValue(unsigned index) const = 0;

virtual int GetCount() const = 0;

virtual DataType GetType() const = 0;

};

IVector-ը թույլ է տալիս ծրագրի իրարից անկախ մոդուլներին միմիանց միջև փոխանակվել տվյալներով միմիայն IVector ինտերֆեյսի ցուցչի միջոցով, առանց մտահոգվելու դրա իրականացման մասին։

Յուրաքանչուր ֆայլ, որը վերլուծության փուլից հետո պետք է ներբեռնվի հիշողության մեջ, ստանում է չկրկնվող տարբերակիչ (identifier), որով էլ ներկայանում է խմբավորման փուլում՝

struct FileInfo

{

int ID;

FileContext m\_fileContext;

QDateTime m\_date;

FileType m\_fileType;

Path m\_filePath;

inline bool operator==(const FileInfo& info)

{

return this->ID == info.ID;

}

};

* FileContext – ֆայլի պարունակության մասին ինֆորմացիա
* ID – չկրկնվող տարբերակիչ
* QDateTime – ֆայլի ստեղծման ժամանակը
* FileType – ֆայլի տեսակը (BinDefinition, Parameterlimits …)
* Path – Ֆայլի գտնվելու վայրը համակարգչի վրա

2.2 Տվյալների խմբավորում

Մեծ քանակի տվյալների հետ աշխատելիս խնդիր է առաջանում տվյալների արագ փնտրման, տվյալների սորտավորման ըստ ինչ-որ պայմանի, որոնք պահանջում են տվյալների պահպանման այնպիսի իրագործում, որի միջոցով արագ կլինի պահանջված տվյալը վերցնելը։ Այս ծրագրի դեպքում իրագործվել է տվյալների խմբավորում ըստ lot, wafer, device պարունակող ֆայլերի, որը թույլ է տալիս ծրագրի մյուս մոդուլներին հաստատուն ժամանակում (const time O(1)) ինֆորմացիա ստանալ ցանկացած lot-ի, wafer-ի, device-ի մասին, քանի որ այս համակարգը իրագործված է QT գրադարանի QHash դասի միջոցով, որն էլ ապահովում է նշված հաստատուն ժամանակում որոնումը։ Իրագործվել է DataIndex դասի միջոցով՝

QHash <QString, IDList> m\_lots;

QHash <QString, IDList> m\_wafers;

QHash <QString, IDList> m\_devices;

Ծրագրի մոդուլից տվյալ վերցնելու պրոցեսը կարելի է նկարագրել հետևյալ դիագրամով՝

Data Provider





2.3 Տվյալների քեշավորում

GetData(FieldList)

FileInfo

Տվյալները ներբեռնման մոդուլը անցնելուց հետո պահվում են DataStore քեշի մեջ, ապահովելով հիշողության օպտիմալ օգտագործում։ Երբ նոր տվյալ վերցնելու հարցում է գալիս ծրագրին, ծրագիրը նախ փնտրում է պահանջվող տվյալին համապատասխան տարբերակիչները (ID) խմբավորված տվյալների մեջից, այնտեղից ստանալով տարբերակիչները, դիմում է իր ներքին քեշին և առկայության դեպքում միանգամից այնտեղից վերցնում պահանջվող տվյալները, անիմաստ ժամանակ չծախսելով վերլուծող և ներբեռնող մոդուլին դիմելու համար։ Քեշավորման բացակայությունը կստիպեր ամեն անգամ դիմել վերլուծող և ներբեռնող մոդուլին, որը պետք է անցներ բոլոր ֆայլերի վրայով համապատասխան տվյալը ներբեռնելու համար, ինչը անտեղի կվատներ ժամանակը։

DataStore դասի հիմնական հաղթաթուղթը QHash<ID,FileData> կառուցվածքն է, որով հնարավորություն է տրվում հաստատուն ժամանակում վերցնել մի ֆայլի ամբողջ պարունակությունը FileData դասի օբյեկտով։

FileData-ն ինքն իրենով ներկայացնում է վերոհիշյալ IVector ինտերֆեյսի ցուցիչների զանգված, քանի որ ֆայլի պարունակութունը սյունակներ են, որոնց ծրագիրը պահում է IVector-ի միջոցով։

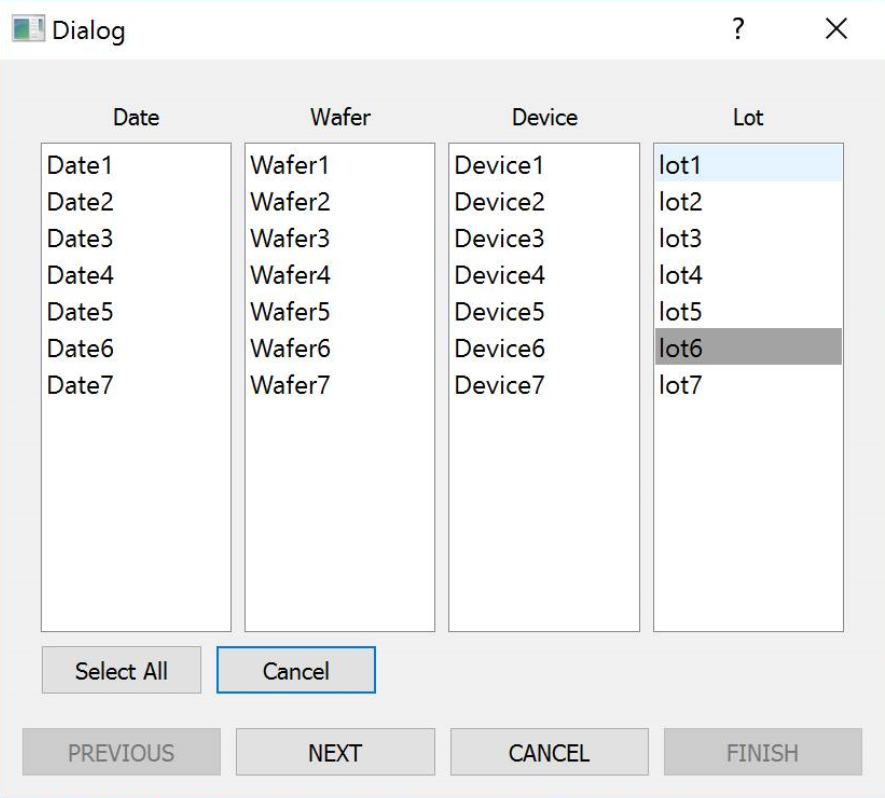
2.4 Սփռված տվյալների հաջորդական ներկայացում

Տվյալները վերլուծության և ներբեռնման փուլը անցնելուց հետո, համակարգչի հիշողության մեջ գտնվում են որոշակի տիրույթներում IVector կառուցվածքների մեջ։ Երբ ծրագրի մի մոդուլը հարցում է անում օրինակ A և B տվյալի համար, իսկ մեկ ուրիշ մոդուլ B և C տվյալի համար, նրանք ստանում են իրանց պահանջած տվյալը, բայց չի կատարվում ընդհանուր տվյալի կրկնօրինակում, նշված օրինակում B տվյալը չի պահվում հիշողության մեջ երկու անգամ։ Կատարվում է տվյալի վիրտուալ ներկայացում IVector ինտերֆեյսի մեկ ուրիշ բնույթի իրականացումների միջոցով, որոնք էլ իրենց հերթին բաժանված են ըստ տվյալի տիպերի՝

* CIntDataGroup
* CDoubleDataGroup
* CStringDataGroup
* CBoolDataGroup
* CDateTimeDataGroup

Այս դասերից յուրաքանչյուրի խնդրին է պահել մի քանի նույն տեսակի IVector-ներ և դրանք ներկայացնել իբրև մեկ ամբողջություն։

2.5 Պայմանին համապատասխան տվյալների վերադարձ

Տվյալները վերադարձնելու համար մշակվել են ITable և IColumn ինտերֆեյսները ու իրենց իրականացումները՝ CTable և CColumn համապատասխանաբար։ ITable-ը իրենից ներկայացնում է IColumn դասի օբյեկտների զանգված, իսկ ահա IColumn-ը կախված տվյալի տեսակից իրենից ներկայացնում է 2.4 գլխում նկարագրված IVector ինտերֆեյսի թվացյալ հաջորդական վեկտորների իրականացումները։Այս պատուհանը տվյալները տվյալների մոդուլից վերցրել է վերոհիշյալ IColumn ինտերֆեյսի միջոցով՝ 

Blank Page

**Գլուխ 3 Տնտեսագիության բաժին**

3.1 Ներածություն, խնդրի դրվածքը

Այս բաժնում նախագծի համար տարվող աշխատանքի առնչությամբ դրվում է էկոնոմիկական հետևյալ խնդիրները`

1. Թեման մշակող ձեռնարկության նկարագիրը,

2.Կազմակերպել այդ աշխատանքները,

3. Կազմել աշխատանքային օրացույցային պլան,

4. Կազմել ծախսերի նախահաշիվ, կատարել վերլուծություն

5. Կազմել արդյունքների հնարավոր իրացման մեխանիզմների նկարագրում:

3.2 Թեման մշակող ձեռնարկության նկարագիրը

Ձեռնարկությունն արտադրական միավոր է, որն ապահովում է ապրանքի կամ ծառայության արտադրություն։ Այս աշխատանքի կատարման համար հիմնադրվում է սահմանափակ պատասխանատվությամբ ընկերություն (ՍՊԸ), որը ընկերության մասնավոր դեպք է և տարբերվում է դասական ընկերությունից նրանով միայն, որ ձեռնարկության պարտքերի դիմաց հիմնադիրները պատասխանատու են միայն իրենց մասնաբաժնով, որևէ այլ բնագավառի այդ "սահմանափակ" տերմինը չի վերաբերում։

Ձեռնարկության աշխատողները օգտագործելով իր տնօրինության տակ գտնվող արտադրության միջոցները, թողարկում են հասարակության համար անհրաժեշտ որևէ տեսակի արտադրանք, ունեն արտադրատեխնիկական, տնտեսական և կազմակերպական միասնություն, օգտվում են  իրավաբանական անձի իրավունքներից և տնտեսությունը վարում են տնտեսավարման որևէ եղանակով:

3.3 Թեմայի մշակման գործընթացի կազմակերպումը: Թեմայի կառուցվածքը Թեման իրականացնող անձնակազմի ձևավորումը

Ավարտական աշխատանքը կատարվելու է մի քանի փուլերով. նախ մշակվելու են գրաֆային որոշակի ալգորիթմներ՝ կատարվելու են ալգորիթմների իրականացման համար նախապատրաստական աշխատանքներ: Այնուհետև ընտրվելու է գրաֆիկական ինտերֆեյսը և թե ինչ ֆունկցիոնալ մասեր է ունենալու ավարտուն աշխատանքը: Հաջորդ փուլում ինտերֆեյսը կապվելու է այդ փուլում նախապես ստեղծված որոշակի ֆունկցիոնալ մասին, իսկ վերջում աշխատանքը բերվելու է վերջնական տեսքի՝ ստեղծվելու են բլոկներ, որոնք ստացված արդյունքներին կտան վերջնական տեսք: Այս փուլերը իրենց հերթին կարելի է բաժանել ենթափուլերի ...

Այս ավարտական աշխատանքի համար հարկավոր են հետևյալ մասնագետները .

* Պրոյեկտի ղեկավար ( մենեջեր ) ( 1 հոգի )
* տեսաբան-մաթեմատիկ ( 1 հոգի )
* ծրագրավորող ( 2 հոգի )

3.4 Աշխատանքների էտապավորումը եվ օրացուցային պլանի կազմումը, աշխատատարությունների որոշումը

Օրացուցային պլանն իր մեջ ներառում է բոլոր այն աշխատանքները, որոնք անմիջականորեն իրականացվում են տվյալ կազմակերպությունում և կողմնակի կազմակերպություններում: Այն մեզ թույլ կտա ռացիոնալ ձևով օգտագործել աշխատանքի կատարման համար տրամադրված ռեսուրսները:

Աշխատանքը կատարվելու է փուլերով, որոնցից յուրաքանչյուրի տևողությունը կլինի.

,

Որտեղ   
-ն -րդ փուլի աշխատատարությունն է ,

- ն -րդ փուլում զբաղված կատարողների թիվն է,

-ն աշխատանքային օրերը օրացուցային օրերի բերող գործակից է. հնգօրյա աշխատանքային շաբաթի դեպքում ,

-ն աշխատանքային օրվա օրական ժամանակային իրական ֆոնդն է

-ն ժամանակային անվանական ֆոնդն է,   
-ն 1 աշխատողի 1 օրվա աշխատաժամանակի կորստի տոկոսը. :  
Կատարվող աշխատանքի փուլերն, անհրաժեշտ աշխատանքային անձնակազմը և օրացուցային պլանի կազմումը ներկայացնենք աղյուսակի տեսքով.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Փուլի անվանումը | Կատարողները | Կատա-րողների թիվը | Օրերի թիվը | Աշխատա-տարությունը |
| 1. | Խնդրի դրվածքի ձևակերպում: | Մենեջեր | 1 | 0.5 | 2.72 |
| 2. | Աշխատանքի մոտավոր  գրաֆիկի կազմում | Մենեջեր | 1 | 0.5 | 2.72 |
| 3. | Թեմայի ուսումնասիրություն | Մենեջեր, մաթեմատիկ | 2 | 3 | 32.64 |
| 4. | Գրականության և լեզվի ընտրություն | Մենեջեր | 1 | 1 | 5.44 |
| 5. | Օրացուցային գրաֆիկի ճշգր-տում, աշխատանքի բաժանում, որին նախորդում է աշխատանքային խմբի ձևավորումը | Մենեջեր | 1 | 3 | 16.32 |
| 6. | Առկա ալգորիթմների ուսում-նասիրություն: Խնդրի լուծման ալգորիթմի ընտրություն: | Մաթեմատիկոս ծրագրավորող | 2 | 3 | 32.64 |
| 7. | Տեխնիկական առաջադրանքի ճշգրտում: | Մենեջեր մաթեմատիկ | 2 | 1 | 5.44 |
| 8. | Ծրագրային միջոցի մշակումը: | Մենեջեր ծրագրավորող   ( 2 հոգի ) | 3 | 10 | 163.2 |
| 9. | Վերջնական արդյունքների ամփոփում և հաշվետվության կազմում | Մենեջեր | 1 | 1 | 2.72 |

8

7

6

5

4

3

2

1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ½ 1 2 4 5 7 8 11 21 22 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

3.5 Թեմայի մշակման ծախսերի նախահաշվի կազմումը եվ վերլուծությունը

Ծախսերի նախահաշվի կազմումն իր մեջ ներառում է հետևյալ ծախսերի հաշվումը:

1. Նյութեր, գնովի պատրաստվածքներ և կիսաֆաբրիկատներ:
2. Հատուկ սարքավորումներ գիտական, փորձարարական աշխատանքների համար:
3. Հիմնական այլ միջոցներ (շենքեր, շինություններ):
4. Գիտաարտադրական անձնակազմի աշխատավարձեր:
5. Սոցապահովագրական հատկացումներ:
6. Գիտաարտադրական գործուղումներ:
7. Կողմնակի կազմակերպություններ, աշխատանքներ և ծառայություններ (հատուկ նպատակային ծախսեր):
8. Այլ ուղղակի ծախսեր:
9. Վերադիր ծախսեր:

Այս բոլոր ծախսերը անհրաժեշտ են, որպեսզի աշխատանքի պլանը կազմվի, հավաքվի որոշակի մասնագետների խումբ և ստեղծվի ծրագրային միջոցը:

3.5.1 Նյութեր, գնովի պատրաստվածքներ և կիսաֆաբրիկատներ

Այս հոդվածում ներառում են բոլոր այն ծախսերը, որոնք ուղղված են նյութերի, կիսաֆաբրիկատների ձեռքբերմանը: Ծախսերի հաշվարկն իրականացվում է ուղիղ ձևով՝ ունենալով նյութի տեսակը, քանակը, միավորի արժեքը, տրանսպորտի տեսակը և ձեռքբերման հումքը:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # | Նյութի անվանումը | Տեսակը | Չափի միավորը | Քանակը | Միավորի գինը | Գումարային արժեքը |
| 1. | Գրենական պիտույքներ | — | Տուփ | 4 | 3000 | 12000 |
| 2. | Խտասկավառակներ | Platinum | Տուփ | 1 | 1000 | 1000 |
| 3. | Տպագրական նյութեր | HP | Հատ | 2 | 7500 | 15000 |
| Ընդամենը | | | | | | 28000 |
| **Ընդամենը՝ հաշվի առած տեղափոխման ծախսերը (7-10%)** | | | | | | **30000** |

3.5.2 Հատուկ սարքավորումներ գիտական, փորձարարական աշխատանքների համար

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | **Հիմնական միջոցի անվանումը** | **Քանակը** | **Միավորի գինը (դր)** | **Գումարը (դր)** | **Նախագծի տևողությունը (օր)** | **Ամորտիզացիայի տևողությունը (տ)** | **Ամորտիզացիոն հատկացումներ** |
| 1. | Համակարգիչ (կոմպլեկտով) | 3 | 250000 | 750000 | 22 | 10 | 4500 |
| 2. | Printer Scanner Copy | 1 | 90000 | 90000 | 22 | 7 | 800 |
| 3. | Գույք | 4 | 60000 | 240000 | 22 | 3 | 5200 |
| 5. | Օդորակիչ | 1 | 200000 | 200000 | 22 | 3 | 4000 |
| **Ընդամենը 1280000** | | | | | | | |
| **Ընդամենը՝ հաշվի առած տեղափոխման ծախսերը (7-10%) 1400000** | | | | | | | |

Այս հոդվածով նախատեսված ծախսերը ներառում են համակարգիչների, հատուկ ստենդերի, արտաքին սարքերի և այլ ձեռքբերումների համար կատարվող ծախսերը: Դրանց համարում են հիմնական միջոցներ և հաշվում են նաև դրանց ամորտիզացիոն ծախսերը նախագծի տևողության ընթացքում:

Սարքավորումների գները որոշվում են տվյալ պահին շուկայում առկա գներով, որին գումարվում են տեղափոխման և մոնտաժման ծախսերը՝ մոտավորապես ապրանքի արժեքի 10 - 15%-ի չափով: Հաշվարկը ներկայացվում է աղյուսակի տեսքով, որի մեջ ամորտիզացիայի նորման որոշվում է ձեռնարկության կողմից:

3.5.3 Հիմնական այլ միջոցներ

Այս հոդվածը ենթադրում է շենքի և շինությունների վրա կատարվող ծախսերի հաշվարկը:  
Աշխատանքը կատարելու համար նպատակահարմար է վարձել որևէ հարմար տարածք: Այդ դեպքում կատարվող ծախսերի հաշվարկը կիրականացվի հետևյալ կերպ. աշխատակազմի համար բավական է 60ք/մ մակերեսով տարածք՝ իր հարմարություններով: Յուրաքանչյուր 1ք/մ-ի տարեկան վարձը 20000 դրամ է, ուստի այդ տարածքի տարեկան վարձը կկազմի 1200000 դրամ: Անհրաժեշտ է վարձակալել 22 աշխատանքային օրով. հետևաբար տարածքը անհրաժեշտ է վարձակալել մեկ մասով և վարձակալման ամբողջ գումարը կկազմի 100000 դրամ:

3.5.4 Գիտաարտադրական անձնակազմի աշխատավարձեր

Այս հոդվածը ներառում է գիտաշխատողների, ինժեներատեխնիկական աշխատակազմի աշխատավարձերը: Աշխատավարձի պլանային ֆոնդի մեծությունը որոշվում է 3 բաղադրիչներով.

Որտեղ -ն հիմնական աշխատավարձ է: Կոչվում է աշխատավարձի տարիֆային ֆոնդ, իսկ -ն պարգևատրումներն են՝ նախատեսված պլանի կատարման, գերակատարման և այլնի համար: Այն պլանավորում են տարիֆային ֆոնդի նկատմամբ 15-20%-ի չափով:

-ըլրացուցիչ աշխատավարձն է (չաշխատած ժամանակի համար): Այն որոշվում է -ի և -ի գումարի 0.02-0.04%-ի չափով:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Փուլի համար | Փուլի աշխատատարությունը | 1մարդ\*ժ-ի արժեքը |  |  |  |  |
| 1. | 2.72 | 3000 | 8160 | - | 340 | 8500 |
| 2. | 2.72 | 3000 | 8160 | - | 340 | 8500 |
| 3. | 32.64 | 3000 | 48960 | 9000 | 1000 | 58960 |
| 2500 | 40800 | 8000 | 1000 | 49800 |
| 4. | 5.44 | 3000 | 16320 | - | 680 | 17000 |
| 5. | 16.32 | 3000 | 48960 | - | 1000 | 49960 |
| 6. | 32.64 | 3000 | 48960 | 6040 | - | 55000 |
| 2500 | 40800 | 8200 | - | 49000 |
| 7. | 5.44 | 3000 | 8160 | - | 340 | 8500 |
| 2500 | 6800 | - | 200 | 7000 |
| 8. | 163.2 | 3000 | 163200 | 24800 | - | 188000 |
| 2200 | 119680 | 20320 | - | 140000 |
| 2200 | 119600 | 20320 | - | 140000 |
| 9. | 2.72 | 3000 | 8160 | - | 340 | 8500 |
| **Ընդամենը 788720** | | | | | | |

3.5.5 Սոցիալապահովագրական հատկացումներ

Այս հոդվածով նախատեսված միջոցները ուղղվում են կենսաթոշակային հիմնադրամ և այլ ապահովագրական հիմնադրամներ: Այն հաշվում են միջինացված գործակցով: Այն կարող ենք հաշվել աշխատավարձի 30%-ի չափով: Այն մոտավորապես կկազմի 240000 դրամ:

3.5.6 Գիտաարտադրական գործուղումներ

Քանի որ աշխատանքի կատարման համար գործուղումների կարիք չկա, ուստի այս  
հոդվածով ոչ մի ծախս չի նախատեսվում:

3.5.7 Կողմնակի կազմակերպություններ, աշխատանքներ և ծառայություններ (հատուկ նպատակային ծախսեր)

Այս ծախսերին դասվում են այլ կազմակերպությունների, ձեռնարկությունների, ինչպես նաև նույն ընկերության այլ ստորաբաժանումների ծախսերը, որոնք տնտեսապես անկախ, ինքնուրույն իրավաբանական անձ են, որոնց մենք պատվիրում ենք որոշակի ծառայություններ՝ թեմայի հետ կապված: Աշխատանքի ընթացքում անհրաժեշտ է օգտվել հեռախոսային և ինտերնետային ծառայություններից:

|  |  |  |  |
| --- | --- | --- | --- |
| # | Ծառայության անվանում | Կազմակերպության անվանումը | Ծառայության գինը (դր) |
| 1. | Հեռախոս | Orange | 20000 |
| 2. | Ինտերնետ | U!com | 30000 |
| **Ընդամենը** | | | **50000** |

3.5.8 Այլ ուղղակի ծախսեր

Այս հոդվածով նախատեսված են մնացած բոլոր այն ծախսերը, որոնք կապված են թեմայի մշակման հետ և աշխատանքի ինքնարժեքի մեջ մտնում են ուղղակիորեն: Այդ ծախսերը իրենց մեջ ընդգրկում են օդորակիչի, համակարգիչների և այլ սարքավորումների էլեկտրաէներգիայի ծախսերը, որոնք օգտագործվում են աշխատանքի կատարման ընթացքում:

15000դր. 15000դր.

3.5.9 Վերադիր ծախսեր

Սրանք այն ծախսերն են, որոնք չեն կարող մտցվել որոշակի թեմայի ինքնարժեքի մեջ: Դրանք կառավարման անձնակազմի ընդհանուր տնտեսական և այլ ոչ արտադրական միջոցառումների համար անհրաժեշտ ծախսերն են: Դրանք հաշվվում են միջինացված գործակցով և կազմում են աշխատավարձի ֆոնդի 50-120%-ը: Հաշվենք այս ծախսերը աշխատավարձի 50%-ի չափով: Այդ գումարը մոտավորապես կկազմի 400000 դրամ:

Կատարվելիք բոլոր ծախսերը ներկայացնենք նախահաշվի տեսքով.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Հոդվածի անվանումը | Ծախսը (դր) | Տոկոսը ընդհանուրի մեջ |
| 1. | Նյութեր, գնովի պատրաստվածքներ և կիսաֆաբրիկատներ | 25000 | 0.8% |
| 2. | Հատուկ սարքավորումներ գիտական, փորձարարական աշ-խատանքների համար | 1400000 | 45.8% |
| 3. | Հիմնական այլ միջոցներ (շենքեր, շինություններ) | 100000 | 3.2% |
| 4. | Գիտաարտադրական անձնակազմի աշխատավարձեր | 788720 | 25.9% |
| 5. | Սոցապահովագրական հատկացումներ | 240000 | 7.9% |
| 6. | Գիտաարտադրական գործուղումներ | — | 0 |
| 7. | Կողմնակի կազմակերպություններ, աշխատանքներ և ծառայություններ (հատուկ նպատակային ծախսեր) | 70000 | 2.3% |
| 8. | Այլ ուղղակի ծախսեր | 30000 | 1% |
| 9. | Վերադիր ծախսեր | 400000 | 13.1% |
| Ընդամենը | | 3053720 | 100% |

Blank page

Գլուխ 4 Կենսագործունեության անվտանգության բաժին

4.1 Հաշվիչ կենտրոններում աշխատանքի պաշտպանությանը ներկայացվող պահանջները

Գիտության և տեխնիկայի զարգացումը հանգեցրել է նրան, որ մշակվել են աշխատանքի կազմակերպման նոր և արդյունավետ տարբերակներ։ Այդ ասպարեզում ամենամեծ նվաճումը եղել է համակարգչի ստեղծումը։ Ներկայումս համակարգիչները զգալիորեն հեշտացնում են մարդու աշխատանքը։ Այդ իսկ պատճառով համակարգիչները շատ լայն կիրառում են գտել անհատ օգտագործողների և ձեռնարկությունների աշխատանքի կազմակերպման մեջ։

Սակայն,բոլոր առավելությունների հետ մեկտեղ, համակարգիչները բացասաբար են ազդում մարդու առողջության վրա։ Այդ պատճառով անհրաժեշտ է մանրամասնորեն ուսումնասիրել համակարգչին ներկայացվող պահանջները, որպեսզի վերջինիս տված վնասները նվազեցվեն, ինչը կբերի աշխատանքի արդյունավետության բարձրացմանը, մարդու առողջության պահպանմանը, անվտանգ և անվնաս աշխատանքային պայմանների ստեղծմանը։

**PC-ների տեղակայման տարածքային պահանջներ** PC-ների և մոնիտորի տեղակայման տարածքները պետք է ունենան բնական և արհեստական լուսավորվածություն։ Բնականն իրականացվում է լուսամուտների միջողով, որոնք ցանկալի է տեղակայել հյուսիսային կամ հյուսիս-արևելյան ճակատներում։ Դա պետք է ապահոցի բնական լուսավորվածության գործակից 1,2 % մշտական ձյունածածկույթով, 1,5 % սովորական վայրերում։ Մեկ մարդուն հատկացվող աշխատանքային մակերեսը պետք է լինի ոչ պակաս, քան 6 մ², իսկ ծավալը՝ ոչ պակաս, քան 20մ³։

Տարածքի ներքին հարդարման համար օգտագործվում են դիֆուզանդրադարձման գործակիցներով՝

առաստաղ` 0.7-0.8, պատեր՝ 0.5-0.6, հատակ՝ 0.3-0.5:

Տարածքի հատակը պետք է լինի հարթ, չսահող, հարմար խոնավ մաքրման համար, ունենա անտիստատիկ հատկություններ։

**Հաշվիչ կենտրոնի աշխատասենյակի լուսավորվածության նշանակությունը աշխատանքի անվտանգության պահպանման հարցում**

Հաշվիչ կենտրոնի աշխատասենյակի լուսավորվածության անթերի կազմակերպման աշխատանքային բարենպաստ պայմանների ստեղծման կարևոր գործոններից է։

Թույլ լուսավորվածությունը ստիպում է մարդուն լարել տեսողությունը, ինչն առաջ է բերում ընդհանուր և, մասնավորապես, տեսողական օրգանների հոգնածություն։ Որքան թույլ է լուսավորվածությունը, այնքան ավելի փոքր է աչքի տեսողական սրությունը, ավելի մանր առարկաներ տարբերելու ունակությունը։ Թույլ լուսավորվածությունը նվազեցնում է նաև տեսողությամբ ընկալելու արագությունը և աչքի կենտրոնային զգացողությունը։

Աչքն ընդունակ է հարմարվելու տվյալ լուսավորվածությանը և պայծառությանը։ Հարմարվելու այդ գործողությունը, որը աչքի ադապտացիա է կողվում, կարող է տևել նույնիսկ մինչև 50 տարի։ Այդ ժամանակահատվածում աչքի տեսողական ունակությունը խիստ նվազում է։ Աչքի հաճախակի ադապտացիան ոչ միայն վնասակար է, այլև լարված աշխատանքում ստեղծում է վտանգավոր պահեր և վթարների պատճառ դառնում։ Անհավասարաչափ կամ թույլ ազդեցությունը տեսողության վրա կարող է նաև տեսողության մասնակի կորստի և մի շարք այլ հիվանդությունների պատճառ դառնալ։

Լուսավորվածությունն ունի նաև գեղագիտական նշանակություն։ Լավ կազմակերպված լուսավորվածության պայմաններում աշխատանքը դյուրին և հաճելի է դառնում։ Հաշվիչ կենտրոնի լուսավորվածությունը պետք է բավարարի հետևյալ պայմաններին

* Լուսավորվածությունը պետք է լինի բավարար և իր սպեկտրով նմանվի բնական լույսին
* Արտադրական շենքում և աշխատատեղերում բարձր ստվերներ կամ կուրացնող պայծառության փայեր չպետք է լինեն
* Լույսի աղբյուրները, ինչպես նաև դրանց համար անհրաժեշտ սարքավորումները պետք է հաճելի տեսք ունենան, լինեն անվնաս և ընտրվեն տնտեսապես նպաստավոր միջոցներով

Լույսը ներկայացվում է 0.380-իվ մինչև 0.760 երկարությամբ էլեկտրամագնիսական ալիքներով, որոնք տարածվում են 3000000 կմ/վ արագությամբ։

0.01-0.38 մկմ երկարությամբ ալիքները կոչվում են ուլտրամանուշակագույն, 0.76-340 մկմ երկարությամբ՝ ինֆրակարմիր ճառագայթներ։ Որպես լույսի հոսք միավոր ընտրվում է լյումենը։ Լույսի տարածման խտությունը՝ լույսի ուժը, չափվում է կանդելներով։

Աշխատատեղերի լուսավորման աստիճանը որոշվում է լուսավորվածությամբ։ Այն ընդունված է չափել հատակի մակերեսից 0.8 մետր բարձրության վրա։ Լուսային զգացողության աստիճանը որոշվում է պայծառությամբ։ Պայծառությունը չափվում է լույսի ուժի և նրա ուղղությանը ուղղահայաց հարթության վրա լուսավորված մակերևույթի պրոեկցիայի հարաբերությամբ։

**Տարածքի լուսավորությանը ներկայացվող պահանջները**

Մոնիտորներով և PC-ներով տարածքները պետք է արհեստականորեն լուսավորվեն համաչափ լուսավորման համակարգով։ Առանձին դեպքերում կարելի է կիրառել տեղային լուսավորում։

Ընդհանուր լուսավորման լամպերի վառությունը, 50 ˚- 90 ˚ ուղղահայացի նկատմամբ ճառագայթման անկյան դեպքում, չպետք է գերազանցի 200 կդ/դմ², իսկ լամպի պաշտպանական անկյունը պետք է լինի մեծ 40 ˚-ից ։

Պուլսացիայի գործակիցը չպետք է գերազանցի 5%-ը, ինչն ապահովում է բարձր հաճախականային գազալիցքավորող լամպերի կիրառմամբ։ Այդ լամպերի բացակայության դեպքում եղած լամպերը պետք է միացնել եռաֆազ ցանցի տարբեր ֆազերին։

PC-ներ և մոնիտորներ պարունակող տարածքները պետք է լինեն այնպիսին, որ բնական լույսն ընկնի կողքից՝ ցանկալի է ձախից։

Աշխատանքային տեղերն պետք է դասավորված լինեն այնպես, որ հաշվի առնվի աշխատանքային սեղանի և վիդեոմոնիտորի միջև հեռավորությունը, որը պետք է մեծ լինի 20 մ-ից, իսկ կողքի վիդեոմոնիտորների միջև՝ 1.2 մ-ից։

Պատուհաններրը պետք է ունենան վարագույրներ, շերտավարագույրներ կամ նմանատիպ հարմարանքներ։

PC-ներ և մոնիտորներ պարունակող տարածքները պետք է անպայման ապահովված լինեն առաջին բուժօգնությանն անհրաժեշտ դեղորայքով։ Հնարավորինս պետք է խուսափել նաև մոնիտորին կողմնակի լույս ընկնելուց. այդ նպատակով համակարգիչը պետք է տեղադրված լինի այնպես, որ դրանից օգտվողի վրա լույսն ընկնի ձախից։ Գործնականում հաճախ խոսելով մոնիտորների վնասակար ազդեցությունների մասին՝ չեն նշվում համակարգչի մյուս բաղադրիչի՝ տպող սարքին առնչվող նման գործոնները։ Մինչդեռ տպող սարքը նույնպես էլեկտրական բարդ սարքավորում է և անվտանգ աշխատելու իր կանոններն ունի։ Նախ՝ պետք է բացառել դրա մեջ պատահական իրեր հայտնվելու վտանգը. այդպիսիք կարող են լինել թղթերի մետաղական ամրուցիչները , և գրասենյակային այլ պիտույքներ։ Սրանք թղթի թերթի հետ հայտնվելով տպող սարքի ներսում՝ վերջինիս շարքից դուրս գալու պատճառ են հանդիսանում։ Լազերային տպող սարքերի աշխատանքը հիմնված է մանրագույն հատիկներով (փոշու հատիկից տասն անգամ փոքր) սև գույնի փոշու, այսպես կոչված՝ տոների վրա։ Տոները խիստ վնասակար, և նույնիսկ՝ թունավոր նյութ է. դրա հատիկները շնչուղիներում կարող են տարբեր հիվանդություններ առաջացնել։ Այդ պատճառով լազերային տպող սարքերի լիցքավորումը պետք է իրականացվի միմիայն մասնագետի կողմից՝ հատուկ պայմաններում։

Անվտանգության կանոնների չպահպանելը կարող է հանգեցնել աշխատակցի աշխատանքի ոչ հարմարավետ լինելուն՝ առաջանում են գլխացավեր, աչքերի ցավեր, հոգնածություն, սրտխառնոց և այլն։ Որոշ մարդկանց մոտ խանգարվում է քունը, ախորժակը, վատանում է տեսողությունը, սկսում են ցավալ ձեռքերը, վիզը, ողնաշարը և այլն։ Կարող են առաջանալ նաև նյարդային խանգարումներ։ Հետազոտությունների արդյունքների շնորհիվ պարզվել է, որ համակարգչով աշխատակցի վատ ինքնազգացողության պատճառներից մեկը հանդիսանում է մոնիտորի էկրանի պատկերի բարձր հաճախականությունը։ Հարկ է նշել, որ՝

* խորհուրդ չի տրվում հերթափոխի ընթացքում աշխատել 6 ժամից ավելի
* խորհուրդ է տրվում համակարգչով աշխատելու ընթացքում կատարել 10 րոպե տևողությամբ ընդմիջում՝ յուրաքանչյուր 50 րոպեից հետո
* համակարգչով անընդմեջ աշխատանքը չկարգավորված ընդմիջման ընթացքում չպետք է գերազանցի 2 ժամին
* ընդմիջումների ընթացքում հարկավոր է կատարել վարժություններ

Blank page

Գլուխ 5 Բնապահպանության բաժին

5․1 Արհեստական աղբյուր ունեցող էլեկտրամագնիսական դաշտերը և նրանց ազդեցությունը շրջակա միջավայրի վրա

Էվոլյուցիայի և կենսագործունեության ընթացքում մարդը փորձարկում է բնական էլեկտրամագնիսական ֆոնի ազդեցությունը, որի բնութագրերը օգտագործվում են որպես ինֆորմացիայի աղբյուր՝ ապահովելով անդադար փոխազդեցությունը արտաքին միջավայրի փոփոխվող պայմանների հետ։ Ժամանակակից հետազոտությունների արդյունքները վկայում են այն մասին, որ բոլոր կենդանի օրգանիզմները, միաբջջից մինչև բարձրագույն կենդանիներ և մարդ, դրսևորում են բացառապես բարձր զգայնություն էլեկտրական և մագնիսական դաշտերի նկատմամբ, որոնց պարամետրերը մոտ են կենսոլորտի դաշտերի բնական պարամետրերին։ Բազմաթիվ վիճակագրական տվյալներով ցույց է տրված, որ բնական աղբյուրների էլեկտրամագնիսական դաշտերը էապես ազդում են կենասաբանական ռիթմերի ձևավորման վրա։ Հայտնաբերվել են բավականին ստույգ փոխադարձ կապեր արևային և գեոմագնիսական ակտիվության և հիպերտոնիկ ճգնաժամերի թվի աճի, սրտամկանի ինֆարկտի, հոգեկան խանգարումների միջև։

Վերջին ժամանակները մարդու և էլեկտրամագնիսական դաշտերի փոխազդեցության խնդիրը դարձել է հրատապ՝ կախված ռադիոկապի և ռադիոլոկացիայի ինտենսիվ զարգացման, տեխնոլոգիական գործողությունների իրականացման համար էլեկրտամագնիսական էներգիայի կիրառության ոլորտի ընդլայնման, կենցաղային էլեկտրական և ռադիոէլեկտրոնային սարքերի մասսայական տարածման վրա։

Եթե դեռ 20-25 տարի առաջ էլեկտրամագնիսական ճառագայթումից պաշտպանվելու խնդիրը վերաբերում էր հիմնականում արտադրական պայմաններին, ապա այսօր բնակչության մեծամասնությունը, փաստորեն, ապրում է արհեստական բնույթի էլեկտրամագնիսական դաշտերում, որոնք օժտված են բավականին բարդ տարածական, ժամանակային և հաճախականային կառուցվածքով։

Արհեստական աղբյուրները ստեղծում են ավելի մեծ ինտենսիվություն էլեկտրամագնիսական դաշտեր, քան բնական աղբյուրները։

Կլինիկա – ֆիզիոլոգիական հետազոտություններով հաստատված է, որ արհեստական ծագումով էլեկտրամագնիսական դաշտերը խաղում են որոշակի դեր սրտանոթային, ուռուցքային, ալերգիկ և արյան հիվանդությունների զարգացման մեջ, ինչպես նաև կարեղ են ազդել գենետիկ կառուցվածքի վրա։

Սիստեմատիկ ազդեցության ժամանակ էլեկտրամագնիսական դաշտերը առաջացնում են արտահայտված փոփոխություններ բնակչության առողջական վիճակի վրա, այդ թվում էլեկտրամագնիսական դաշտերի աղբյուրների հետ մագնիսորեն կապ չունեցող անձանց վրա, ընդ որում, թույլ ինտենսիվության դաշտերի ազդեցության էֆեկտը կրում է հեռակա բնույթ։ Բացահայտված է նյարդային համակարգի, ակնաբյուրեղի տղամարդկանց սերմնային գեղձերի խոցելիությունը և բարձր զգայնությունը, հայտնաբերվել են ներզատական ապարատի բոլոր օղակների ֆունկցիոնալ կառավարման, ճարպային փոխանակության և մի շարք այլ խախտումներ։ Աշխատանքների զգալի մասը վկայում են գենետիկական կառուցվածքի, բջջաթաղանքի, իմունային համակարգի, հորմոնալ ստատուսի վրա էլեկտրամագնիսական դաշտերի բացասական ազդեցության մասին։Վերջին տարիների հրատարակություններում ակտրվորեն քննարկվում է այսպես կոչված «արտադրական» հաճախության (50 Հց Ռուսաստանում, Եվրոպայում և 60 Հց Ամերիկայում) էլեկտրամագնիսական դաշտերի քաղցկեղածին վտանգի հարցը։

Անտրոպոգեն աղբյուրների էլեկտրամագնիսական ճառագայթումները ներկայացնում են մեծ բարդություններ ինչպես անալիզի, այնպես էլ ճառագայթման ինտենսիվության սահմանափակման տեսանկյունից։ Դա պայմանավորված է հետևյալ հիմնական պատճառներով

* մեծամասամբ հնարավոր չէ սահմանափակել աղտոտող գործոնի արտանետումները շրջակա միջավայր
* տվյալ գործոնի փոխարինումը մեկ այլ, ավելի քիչ վտանգավոր գործոնով հնարավոր չէ
* հնարավոր չէ եթերի «մաքրումը» անցանկալի ճառագայթումներից
* անընդունելի է մեթոդական մոտեցումը
* հավանական է էլեկտրամագնիսական դաշտի երկարաժամկետ ազդեցություն
* հնարավոր է ազդեցություն մեծ թվով մարդկանց վրա, այդ թվում երեխաների, ծերերի և հիվանդների վրա

Վերջին ժամանակները էլեկտրամագնիսական անվտանգության խնդիրը ձեռք է բերում սոցիալական նշանակություն։ Իրավիճակը բարդանում է նրանով, որ մարդու զգայական օրգանները, բացառությամբ որոշ դեպքերի, մինչև տեսանելի միջակայքի հաճախության էլեկտրամագնիսական դաշտերը չեն ընկալում, ինչի պատճառով, առանց համապատասխան սարքերի, հնարավոր չէ գնահատել ճառագայթման վտանգի աստիճանը։

Ցածր հաճախականային էլեկտրամագնիսական դաշտեր էլէկտրաստատիկ լիցքերով մակերևույթներ

Կենցաղային պայմաններում էլեկտրաստատիկ դաշտերի աղբյուրները կարող են հանդիսանալ ցանկացած մակերևույթներ և առարկաներ, որոնք հեշտությամբ լիցքավորվում են շփման ժամանակ․ գորգեր, լաքապատված ծածկույթներ, սինթետիկ գործվածքներից շորեր։ Բացի դրանից, էլեկտրաստատիկ լիցքեր են կուտակվում հեռուստացույցների էլեկտրոնաճառագայթային խողովակի, օսցիլոգրաֆի վրա։

Էլեկտրաստատիկ դաշտերի լարվածությունները բնակելի շենքում կարող են կազմել 20..40 Կվ/մ։

Էլեկտրահաղորդման օդային գծեր

Տվյալ աղբյուրի էլեկտրամագնիսական դաշտի ինտենսիվությունը մեծամասամբ կախված է գծի լարումից (110,220,330 Կվ և բարձր)։ Էլեկտրամոնտյորների աշխատանքային վայրերում լարվածության միջին արժեքներն են E = 5…15 Կվ/մ, H=1…5 Ա/մ, սպասարկվող անձնակազմի շրջանցման ուղիներում՝ E = 5…30 Կվ/մ, H=2…10 Ա/մ։ Բնակելի շենքերում, որոնք տեղակայված են բարձրավոլտ գծերի մոտակայքում, էլեկտրական դաշտի լարվածությունը որպես կանոն, չի գերազանցում 200-300 Վ/մ, իսկ մագնիսական դաշտինը՝ 0.2…2Ա։

Մեր երրում արտադրվող և արտասահմանից ներմուծվող բարձր հաճախականային վառարանները աշխատում են 2450 ՄՀց հաճախությամբ։ Այդպիսի սարքավորումների մագնետրոն գեներատորների տատանվող հզորությունը կախված է վառարանի ծավալից և կարող է հասնել մինչև 800 Վտ։

Էլեկտրամագնիսական էներգիայի ճառագայթումը շրջակա միջավայր մեծամասամբ պայմանավորված է տեխնոլոգիական անսարքություներով և խախտումներով (օրինակ՝ ոչ լրիվ փակված դռներ)։ Անսարք վառարանների դիտարկումները ցույց են տվել, որ էներգիայի հոսքի խտության առավելագույն արժեքը վառարանից 5 սմ հեռավորության վրա կազմում է 100 ՄՎտ/սմ։

Կլանիչ հատկություններով և մեխանիկական ճկունությամբ օժտված պոլիմերային, ֆեռոմագնիսական նյութերի օգտագործումը հանդիսանում է նոր ուղղության գերբարձր հաչախականային վառարանների արտադրության մեջ։ Այս նյութերը թույլ են տալիս ապահովել էկրանավորող և ռադիոհերմիտիզացնող նյութերի կիպ հպումը իրանին կամ միացումներին էկրանավորման բարձր գործակցի պայմաններում։

Ռադիոհաղորդող սարքեր

Ռադիոհաղորդող սարքերը, որոնք օգտագործվում են ռադիոտեղորոշման, ռադիոնավիգացիայի և կապի համար, աշխատում են շատ լայն հաճախականային միջակայքում 9 կՀցից մինչև հարյուրավոր գեգահերցեր։ Հաղորդվող ալեհավաքների ճառագայթող հզորությունները նույնպես շատ բազմազան են։

Էլեկտրոնաճառագայթային խողովակները հանդիսանում են լայն հաճախականային միջակայքի էլեկտրամագնիսական ճառագայթումների աղբյուրներ։ Էլեկտրոնաճառագայթային խողովակների առաջացրած ցածրհաճախականային, բարձրհաճախականային, ինֆրակարմիր, տեսանելի լուսային, ուլտրամանուշակագույն և ռենտգենյան ճառագայթումները պահանջում են հատուկ անալիզ և յուրահատուկ պաշտպանական միջոցառումների կազմակերպում։ Ցածր հաճախականային և բարձր հաճախականային միջակայքերում էլեկտրամագնիսական ճառագայթումների հիմնական աղբյուրները հանդիսանում են

* մոնիտորի էկրանը
* սնման գծերը և համակարգային բլոկը
* տողային տեսածրման համակարգը
* կադրային տեսածրման համակարգը

Համեմատաբար վերջերս առաջացել է էլեկտրամագնիսական դաշտերի նոր աղբյուր՝ իմպուլսային սնման բլոկը։ Ցանցային տրանսֆորմատորների գաբարիտային չափսերի և քաշի փոքրացման համար նաև սնման լարման հաճախությունը բարձրացնում են մինչև 100…150 կՀց, հետո նոր տրանսֆորմացնում են արդեն այդ հաճախության վրա։ Ճառագայթման ամենաուժեղ մակարդակները նկատվում են մոնիտորի վերին և կողային պատերից։

Էլեկտրամագնիսական անվտանգւթյան տեսանկյունից հատուկ ուշադրության են արժանի notebook տեսակի համակարգիչները։ Նրանցում բացակայում է բարձրավոլտ տեսածրման տողերի բլոկները և գումարային ճառագայթումը գրեթե ամբողջությամբ պայմանավորվում է իմպուլսային սնման բլոկներով։ Այդպիսի բլոկները մի քանիսն են՝ ցանցային ձայնարկիչ, էլեկտրոնիկայի սնման բլոկ, ներքին լուսավորվող հարթ էկրան։ Միջազգային կազմակերպություններից,որոնք մշակում են նորմեր և հանձնարարականներ այս բնագավառում, հարկ է նշել

* Առողջապահության Միջազգային Կազմակերպությունը
* Միջազգային Էլեկտրոնային հանձնաժողովը
* Նեոնիզացնող Ռադիացիայից Պաշտպանության Միջազգային Կոմիտեն

Ներկայումս Եվրոխորհուրդում տարվում են էլեկտրամագնիսական դաշտերի պարամետրերի հիգիենիկ նորավորման բավականին ակտիվ աշխատանքներ։

Եզրակացություն

Աշխատանքի ընթացքում իրականացվել է տվյալների պահպանման կառուցվածքների նախագծում, որոնք արդյունավետ կերպով են օգտագործում համակարգչի հիշողությունը՝ քեշավորելով և խմբավորելով արդեն իսկ ներբեռնված տվյալները և իրականում գտնվելով հիշողության մեջ սփռված, օգտագործողին ներկայանում են թվացյալ կերպով հաջորդաբար։

Օգտագործված գրականության ցանկ

1. Mark Summerfield - Creating Great Software with C++/Qt 4 2010
2. Stroustroup - The C++ Programming Language 4th edition 2012
3. Mark Allen -Data structures and algorithm analysis in C++ 2006
4. Магеррамов - Процесс тестирования интегральных микросхем 2015