Mesajları iletmenin ÖNEMLİ BİR YÖNTEMİ, yerlerine sembol dizilerini iletmektir. Mevcut sembol türlerinden daha fazla gönderilebilecek mesaj varsa, bazı mesajların birden fazla sembol kullanması gerekir. Her sembolün iletim için aynı zamana ihtiyaç duyduğu varsayılırsa, bir mesajın iletim süresi (uzunluğu), onunla ilişkili sembollerin sayısı ile doğru orantılıdır. Bu yazıda, verilen bir mesajla ilişkilendirilen sembol veya semboller dizisine "mesaj kodu" adı verilecektir. İletilebilecek tüm mesaj sayısı "mesaj topluluğu" olarak adlandırılacaktır. Topluluğun her mesajı için kodun anlamı hakkında verici ve alıcı arasındaki karşılıklı anlaşmaya "topluluk kodu" adı verilir.

Muhtemelen en tanıdık topluluk kodu, "karadan bir ise ve denizden iki ise" ifadesinde belirtilmiştir. Bu durumda, mesaj topluluğu "karadan" ve "deniz yoluyla" iki ayrı mesajdan oluşuyordu ve mesaj kodları "bir" ve "iki" idi.

Bir topluluk kodunun gerekliliklerini resmileştirmek için kodlama sembolleri sayılarla temsil edilecektir. Böylece, kodlamada kullanılacak D farklı tipte sembol varsa, bunlar 0, 1, 2, rakamları ile temsil edilecektir. . , (D-1). Örneğin, 0, 1 ve 2 rakamlarını kodlama sembolleri olarak kullanarak üçlü bir kod oluşturulacaktır.

Gruptaki mesaj sayısı N olarak adlandırılacaktır. P(i) i. mesajın olasılığı olsun. Sonra

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bir mesajın uzunluğu, L(i), kendisine atanan kodlama basamaklarının sayısıdır. Bu nedenle, ortalama mesaj uzunluğu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

"Artıklık" terimi, Shannon tarafından kodların bir özelliği olarak tanımlanmıştır. burada, sonlu sayıda üyeden (N) ve belirli sayıda kodlama basamağından (D) oluşan bir mesaj topluluğu için, mümkün olan en düşük ortalama mesaj uzunluğunu veren bir topluluk kodu olarak tanımlanacaktır. Uzun "minimum fazlalık" teriminin kullanılmasından kaçınmak için, bu terim burada "optimum" ile değiştirilecektir. O zaman bu yazıda "optimum kod"un "minimum fazlalıklı kod" anlamına geldiği anlaşılacaktır. Bir topluluk koduna aşağıdaki temel kısıtlamalar uygulanacaktır:

(a) Hiçbir iki mesaj aynı kodlama rakamlarından oluşmayacaktır.

(b) Mesaj kodları, bir mesaj dizisinin başlangıç noktası bilindiğinde, bir mesaj kodunun nerede başlayıp nerede bittiğini belirtmek için ek bir gösterge gerekmeyecek şekilde oluşturulacaktır.

Kısıtlama (b), hiçbir mesajın, kodunun, daha büyük uzunluktaki herhangi bir mesaj kodunun ilk parçası olarak, basamak basamak görünecek şekilde kodlanmamasını gerektirir. Bu nedenle, 01, 102, 111 ve 202, dört üyeli bir topluluk için geçerli mesaj kodlarıdır. Örneğin, bu mesajların 1111022020101111102 dizisi, 111-102-202-01-01-111-102 ayrı mesajlara bölünebilir. Alıcının bilmesi gereken tek şey topluluk kodudur. Bununla birlikte, topluluğun 11, 111, 102 ve 02 dahil olmak üzere bireysel mesaj kodları varsa, o zaman bir mesaj dizisi 11 rakamlarıyla başladığında, mesajın 11 alınıp alınmadığı veya sadece ilk ikisinin olup olmadığı hemen kesin değildir. Ayrıca, dizi 11102 olarak çıksa bile, 111-02 mi yoksa 11-102 mi iletildiği hala kesin değil. Bu örnekte, iki mesaj kodundan (111 veya 11) birinin değişmesi belirtilmiştir.

C. E. Shannon' ve R. M. Fano2, mesaj başına gereken ortalama ikili basamak sayısının, mesaj başına ortalama bilgi miktarının üzerinde olduğunu kanıtlamak amacıyla topluluk kodlama prosedürleri geliştirmiştir. Kodlama prosedürleri optimum değildir, ancak N sonsuza yaklaştığında optimum davranışa yaklaşır. Grup sonlu sayıda üye içerdiğinde ideale mümkün olduğunca yakın bir ortalama kod uzunluğu veren bir kodlama yöntemi geliştiren Krnft3 townrcl tarafından bazı çalışmalar yapılmıştır. Ancak, şimdiye kadar, yazarın bilgisine göre böyle bir kodun oluşturulması için kesin bir prosedür önerilmemiştir. Böyle bir prosedür türetmek bu makalenin amacıdır.

**DERIVED CODING REQUIREMENTS (TÜREVLİ KODLAMA GEREKSİNİMLERİ)**

Optimum bir kod için, verilen bir mesaj kodunun uzunluğu asla daha olası bir mesaj kodunun uzunluğundan daha az olamaz. Bu gereklilik karşılanmazsa, söz konusu iki mesajın kodlarını, daha kısa olan kod daha olası mesajla ilişkilendirilecek şekilde değiştirilerek ortalama mesaj uzunluğunda bir azalma elde edilebilir. Ayrıca, aynı olasılığa sahip birden fazla mesaj varsa, bu mesajların kodlarının uzunlukları farklı olabilir. Ancak, bu mesajların kodları, mesaj grubu için ortalama kod uzunluğunu etkilemeden herhangi bir şekilde değiştirilebilir. Bu nedenle, topluluktaki mesajların şu şekilde sıralandığı varsayılabilir:



ve buna ek olarak, optimum bir kod için koşul



tutar. Bu gereksinimin aşağıdaki tartışma yoluyla karşılandığı varsayılmaktadır.

Bir topluluk kodunun N. mesaja (N-l)st mesajdan q daha fazla rakam atayabileceği düşünülebilir. Ancak N. mesajın ilk L(N- 1) hanesi başka bir mesaj için kod olarak kullanılmamalıdır. Böylece ek q rakamları hiçbir yararlı amaca hizmet etmeyecek ve Lav'ı gereksiz yere artıracaktır. Bu nedenle, optimum bir kod için L(N)'nin L(N-1)'e eşit olması gerekir.

Bir mesaj kodunun k. öneki, o mesaj kodunun ilk k hanesi olarak tanımlanacaktır. Temel kısıtlama (b) daha sonra şu şekilde yeniden ifade edilebilir: Hiçbir mesaj, kodu başka bir mesajın öneki olacak veya öneklerinden herhangi biri başka bir yerde mesaj kodu olarak kullanılacak şekilde kodlanmamalıdır.

L(N) uzunluğu ile kodlanmış iki mesajın hiçbirinin L(N) -1 sıralı aynı öneklere sahip olmadığı bir optimum kod düşünün. Optimum bir kod varsayıldığından, L(N) uzunluğundaki bu mesajlardan hiçbiri, diğer kodlara karşılık gelen herhangi bir sıradaki kodlara veya öneklere sahip olamaz. O zaman tüm bu mesaj grubunun son rakamını düşürmek ve böylece Lav'ın değerini düşürmek mümkün olacaktır. Bu nedenle, bir optimum kodda, L(N) uzunluğundaki kodların en az ikisinin (ve D'den fazlasının) aynı L(N)-1 sıra ön eklerine sahip olması gerekir.

Optimum bir kod için ek bir gereklilik yapılabilir. Uzunluğu L(N) rakamından daha kısa olan ve mesaj kodu olarak kullanılmayan veya bir mesaj kodunun öneki olmayan D farklı tipte kodlama rakamlarının bir kombinasyonunun var olduğunu varsayın. Daha sonra, bu rakam kombinasyonu, N. mesajın kodunu La'nın müteakip azalmasıyla değiştirmek için kullanılabilir. Bu nedenle, tüm olası L(N)-1 rakam dizileri ya mesaj kodları olarak kullanılmalı ya da bunlardan birine sahip olmalıdır. mesaj kodu olarak kullanılan önekler.

Optimum bir kod için türetilen kısıtlamalar, bu belgenin ilk bölümünde verilen (a) ve (b) kısıtlamalarına ek olarak dikkate alınan ve aşağıda özetlenmiş biçimde özetlenmiştir:

(c) L(1) < L(2) < ... < L(N-;) L(N). (5)

(d) Kod uzunluğu L(N) olan mesajların en az ikisi ve en fazla tP'si, son rakamları dışında benzer kodlara sahiptir.

(e) Her olası L(N) -1 basamak dizisi ya bir mesaj kodu olarak kullanılmalı ya da öneklerinden biri mesaj ccda olarak kullanılmalıdır.

**OPTIMUM BINARY CODE (OPTİMUM İKİLİ KOD)**

Optimum kodlama prosedürünü geliştirme kolaylığı için, şimdi kendimizi ikili kodlama problemi ile sınırlayalım. Daha sonra bu prosedür, genel D rakamı durumuna genişletilecektir.

Kısıtlama (c), olası en düşük iki mesajın eşit uzunlukta kodlara sahip olmasını gerekli kılar. Kısıtlama (d), D'nin ikiye eşit olması için, kod uzunluğu L(N) olan ve son basamakları dışında aynı olan mesajlardan sadece ikisinin olması gerekliliğini getirir. Bu iki kodun son haneleri iki ikili haneden biri olacaktır, 0 ve 1. L(N) uzunluğunda başka kodlar mevcut değildir. Bu yapıldıktan sonra, bu iki mesaj tek bir bileşik mesaja eşdeğerdir. Kodu (henüz belirlenmemiş), bu iki mesajın L(N) -1 sırasının ortak önekleri olacaktır. Olasılığı, oluşturulduğu iki mesajın olasılıklarının toplamı olacaktır. İki bileşenli mesajların yerine bu birleşik mesajı içeren topluluğa, birinci yardımcı mesaj topluluğu adı verilecektir.

Bu yeni oluşturulan topluluk, orijinalinden on6 daha az mesaj içeriyor. Gerekirse üyeleri yeniden düzenlenerek mesajların olasılıklarına göre yeniden sıralanması gerekir. Orijinal topluluk olduğu gibi tam olarak kabul edilebilir. Bu yeni gruptaki olası en düşük iki mesajın her birinin kodlarının, son rakamları dışında aynı olması gerekir; 0 ve 1, iki mesajın her biri için birer tane olmak üzere bu rakamlar olarak atanır. Her yeni yardımcı topluluk, önceki gruptan bir eksik mesaj içerir. Her bir yardımcı grup, birikmiş gerekli kodlama gereksinimlerinin tam kullanımıyla orijinal topluluğu temsil eder.

İşlem, en son oluşturulan yardımcı mesaj topluluğundaki üye sayısı ikiye düşene kadar tekrar tekrar uygulanır. Bu iki bileşik mesajın her birine ikili rakamlardan biri atanır. Bu mesajlar daha sonra olasılık birliği olan tek bir bileşik mesaj oluşturmak üzere birleştirilir ve kodlama tamamlanır.

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şimdi Tablo I'i inceleyelim. Soldaki sütun, kodlanacak topluluğun sıralı mesaj olasılıklarını içerir. Nis 13'e eşittir. İki mesajın (bir parantez ile gösterilen) her kombinasyonuna, her birine yeni bir rakam atanması eşlik ettiği için, her bir orijinal mesaja atanması gereken toplam rakam sayısı, mesaj sayısı ile aynıdır. bu mesaj için belirtilen kombinasyonlar. Örneğin, $ işaretli mesaj veya parçası olduğu bir bileşik, diğerleriyle beş kez birleştirilir ve bu nedenle, beş basamaklı bir kod uzunluğu atanmalıdır.

Olasılığı en düşük olan iki mesajı seçmenin bir alternatifi olmadığında, gerektiği gibi belirlenen gereksinimlerin de optimum bir kod elde etmek için yeterli olduğu açıktır. En az olası mesajların iki veya daha fazla gruplaması arasında seçim yapılabileceği durumlar ortaya çıkabilir. Böyle bir durum, örneğin, Tablo I'in dördüncü yardımcı grubunda ortaya çıkar. 0,08 olasılık mesajlarından herhangi biri 0,06 olasılık mesajı ile birleştirilebilirdi. Ancak, ortalama kod uzunluğunu etkilemeden eşit olasılığa sahip mesajlar arasında kodları herhangi bir şekilde yeniden düzenlemek mümkündür ve bu nedenle alternatiflerden herhangi biri yapılabilirdi. Bu nedenle, verilen prosedür her zaman optimum bir ikili kod oluşturmak için yeterlidir. Tablo I'den türetilen tüm kodlanmış mesajların uzunlukları Tablo II'de verilmektedir.

Artık her mesaj için uygun kod uzunluklarını belirledikten sonra, gerçek rakamları belirleme sorunu devam ediyor. Birçok alternatif mevcuttur. Mesajların bileşimlerinde birleştirilmesi, damlamaların, derelerin, derelerin ve derelerin son büyük bir nehirde birbirini takip eden birleşimlerine benzer olduğu için, şimdiye kadar açıklanan prosedür, suda taşınan bir böcek tarafından işaretlerin yerleştirilmesine benzer olarak düşünülebilir. bu kavşakların her biri aşağı yönde yolculuk ederken. Arzuladığımız kodun, böceğin akıntıya karşı geri dönebilmesi için hatırlaması gereken kod olduğu unutulmamalıdır. İşaretlerin yerleştirilmesi, her kavşakta "sıfır-sağa dönüş" gibi aynı kuralı takip etmek zorunda olmadığı için, örneğimiz için kod basamaklarını atamanın en az 212 farklı yolu olduğu görülebilir.

tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tablo II'deki kod, herhangi bir parantezin üst mesajı için 0 rakamı ve alt mesaj için 1 rakamı kullanılarak elde edilmiştir. Tablo I'de, her bir parantez içine iki mesaj (bir değil) yerleştirildiği sürece kodlama kısıtlamasının (e) otomatik olarak karşılandığını belirtmek önemlidir.

Üç veya daha fazla sayı tipi kullanan bir mesaj grubunun optimum kodlaması, ikili kodlama prosedürüne benzer. Tablo I'e benzer bir yardımcı mesaj toplulukları tablosu kullanılacaktır. Birleşik mesajları oluşturmak için birleştirilen mesajları gösteren parantezler, Tablo I'dekiyle aynı şekilde kullanılacaktır. Ancak, kısıtlamayı (e) karşılamak için, olası bir istisna dışında, tüm bu parantezlerin kullanılması gerekecektir. Orijinal grubun en az olası mesajları, her zaman D'ye eşit sayıda mesajı birleştirir.

Sonlandırıcı yardımcı topluluğun her zaman bir birlik olasılık mesajına sahip olduğu not edilecektir. İlk yardımcı topluluğa ulaşılana kadar önceki her grubun sayısı D -1 kadar artırılır. Bu nedenle, N1, birinci yardımcı topluluktaki mesaj sayısıysa, (N1-1)/(D-1) bir tamsayı olmalıdır. Ancak Ni= NI\-no+ 1, burada hayır, orijinal toplulukta bir parantez içinde birleştirilen en düşük olası mesajların sayısıdır. Bu nedenle, no (elbette en az iki ve D'den fazla olmayan), (N-no)/(D-1) bir tam sayı olacak şekilde bir değerde olmalıdır.

Tablo III'te, dört rakamla kodlanacak sekiz mesajdan oluşan bir grubun kullanıldığı bir örnek ele alınmaktadır; nO 2 olarak bulunur. Tablodaki kod, parantezlerin her birine sırasıyla 0, 1, 2 ve 3 rakamlarının atanmasıyla elde edilir.

metin, makbuz, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

**TEŞEKKÜRLER (ACKNOWLEDGMENTS)**

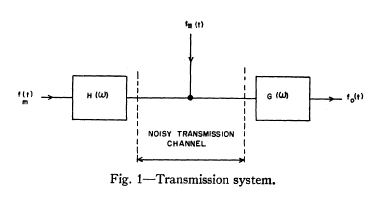
Yazar, her ikisi de Massachusetts Teknoloji Enstitüsü'nden Dr. W. K. Linvill ve Dr. R. M. Fano'ya bu makaleye yönelik yararlı eleştirileri için minnettardır.

**Coding with Linear Systems\* (Lineer Sistemlerle Kodlama\*)**

**Özet-Gürültülü bir kanal üzerinden mesaj iletimi kabul edilir. İki lineer ağ tasarlanacaktır: biri iletimden önce mesajı işlemek için kullanılacak ve ikincisi işlenen mesajı artı alıcı uçtaki kanal gürültüsünü filtrelemek için kullanılacaktır. Gerçek iletim devresi çıkışı ile gecikmeli mesaj arasındaki ortalama kare hatası, uygun ağ tasarımı ile belirli bir izin verilen ortalama sinyal gücü için en aza indirilir. Sayısal örnekler verilir ve tartışılır.**

**I. INTRODUCTION**

Burada dikkate alınması gereken SORUN, gürültülü bir kanal üzerinden mesaj iletimidir. Şekil l'de gösterildiği gibi, fm(t) mesaj fonksiyonu, bir gürültü fonksiyonunun, fn(t) ek olarak tanıtıldığı bir kanala gönderilmelidir. Elde edilen sistem çıktısı, fo(t) ile temsil edilir. Çoğu iletişim sisteminde, mesajı iletim kanalına girmeden önce kodlama fırsatı vardır. Son zamanlarda, Wiener, Shannon ve diğerleri, mesaj fonksiyonunun örneklendiği, nicelendiği ve elde edilen örnek değerlerin iletim için bir darbe koduna dönüştürüldüğü, oldukça karmaşık bir yapıya sahip kodlama süreçlerini düşündüler. Bu teknik birçok durumda oldukça faydalı olsa da, uygulanması gereken uçbirim ekipmanının karmaşıklığı nedeniyle kısıtlanacaktır. Bu tartışmada, kodlama ve kod çözme sistemleri doğrusal ağlarla sınırlandırılacaktır. Şekil 1'de, iletimden önce mesajı kodlamak için H(w) ağı kullanılacak ve G(c) ağı gerekli kod çözme işlemini gerçekleştirecektir. H(w) ağı, mesajın önceden çarpıtılacağı veya kod çözme veya filtreleme ağının G(w)'nin mesajın kendisi olmadan gönderilmiş olsaydı mümkün olandan daha iyi bir sistem çıktısı vermesini sağlayacak şekilde kodlanacağı şekilde tasarlanmalıdır değişiklik.



Daha ileri gitmeden önce, iletim sistemi için bir performans kriteri seçilmelidir. Yani, belirli bir H(w) - G(w) ağ çiftinin diğer bir çiftten daha tatmin edici olup olmadığını belirlememizi sağlamak için bazı ölçülebilir niceliklere karar verilmelidir. Tüm durumlarda tek bir performans kriterinin uygulanması beklenemez ve kullanılacak performansın ortalama hata ölçüsü için böyle bir iddiada bulunulmaz.

Yazar, sonuçta tek bir hata olmadan çok sayıda işlemin gerçekleştirilmesi gereken büyük ölçekli bilgisayar makinelerini dikkate alarak bu makalede verilen çalışmaya yönlendirilmiştir. Büyük ölçekte "işleri doğru yapma" sorunu aslında yeni değil; örneğin bir telefon santralinde çok sayıda işlem yapılırken, yanlış numaralara yol açan hatalar tamamen ortadan kaldırılamasa da kontrol altında tutulur. Bu, kısmen kendi kendini kontrol eden devrelerin kullanılmasıyla başarılmıştır. Rutin kontrolden kaçan ara sıra arıza, müşteri tarafından hala tespit edilir ve devam ederse müşteri şikayetiyle sonuçlanırken, geçici ise yalnızca ara sıra yanlış numaralar üretecektir. Aynı zamanda, merkez ofisin geri kalanı tatmin edici bir şekilde çalışır. Öte yandan, dijital bir bilgisayarda, tek bir arıza genellikle tam arıza anlamına gelir, yani tespit edilirse arıza bulunup düzeltilene kadar daha fazla bilgi işlem yapılamaz, tespitten kaçarsa, o zaman hepsini geçersiz kılar. makinenin sonraki işlemleri. Başka bir deyişle, bir telefon merkez ofisinde, birbirinden az çok bağımsız olan birkaç paralel yol vardır; dijital bir makinede, genellikle, cevap alınmadan önce aynı ekipman parçasından birçok kez geçen tek bir uzun yol vardır.

Bilgiyi bir yerden başka bir yere aktarırken, dijital makineler, anlamların veya değerlerin eklendiği basit semboller kümesi olan kodları kullanır. İzole edilmiş hataları tespit etmek için tasarlanmış kod örnekleri çoktur; bunların arasında, yaygın olarak kullanılan kontrol anahtarlama sistemlerinde ve Bell Röle Bilgisayarlarda yaygın olarak kullanılan 5 koddan 2'si, "radyo telgrafı için kullanılan 7 koddan 3'ü" ve telgrafların sonunda gönderilen kelime sayısı vardır.

Bazı durumlarda kendi kendine kontrol yeterli değildir. Örneğin, Bell Telephone Laboratories tarafından Aberdeen Proving Grounds için inşa edilen Model 5 Röle Bilgisayarlarında, erken dönemdeki gözlemler, iki bilgisayarın 8900 rölesinde günde yaklaşık iki veya üç röle arızası olduğunu gösteriyordu, bu da her iki bilgisayar için bir arızayı temsil ediyordu. üç milyon röle işlemine kadar. Kendi kendini kontrol etme özelliği, bu arızaların tespit edilmemiş hatalara yol açmaması anlamına geliyordu. Bununla birlikte, makineler, geceler ve hafta sonları boyunca gözetimsiz olarak çalıştırıldığından, hatalar, makineler genellikle yeni problemler üstlense de, hesaplamaların sıklıkla durma noktasına geldiği anlamına geliyordu. Mevcut eğilim, temel öğelerin işlem başına rölelerden biraz daha güvenilir olduğu dijital bilgisayarlarda elektronik hızlara yöneliktir. Bununla birlikte, tespit edildiğinde bile izole arızaların görülme sıklığı, bu tür makinelerin normal kullanımını ciddi şekilde etkileyebilir. Bu nedenle, hata tespitinin ötesindeki bir sonraki adımı, yani hata düzeltmeyi incelemek arzu edilir görünmektedir.

İletim ekipmanının bilgiyi 0'lar ve 1'ler dizisinin ikili biçiminde ele aldığını varsayacağız, bu nedenle bu varsayım hem matematiksel kolaylık için hem de ikili sistem açık ve kapalı röleleri, flip-flop'u temsil etmenin doğal biçimi olduğu için yapılmıştır. birçok iletişim biçiminde kullanılan devreler, noktalar ve tireler ve delikli bantlar. Böylece her kod sembolü, 0'lar ve 1'ler dizisi ile temsil edilecektir.

Bu yazıda kullanılan kodlara sistematik kodlar denir. Sistematik kodlar", her bir kod sembolünün tam olarak n ikili basamağa sahip olduğu, m basamak bilgi ile ilişkilendirilirken diğer k = n - m basamak hata tespiti ve düzeltme için kullanıldığı kodlar olarak tanımlanabilir. Bu, R tanımlı bir fazlalık üretir. oranı olarak

aynı bilgiyi iletmek için gereken minimum sayıda kullanılan ikili basamak sayısı, yani R = n/m.

Bu, bilgi aktarımı söz konusu olduğunda kodun etkinliğini ölçmeye hizmet eder ve burada ayrıntılı olarak tartışılan sorunun tek yönüdür. Fazlalığın, bilgi göndermek için etkin kanal kapasitesini düşürdüğü söylenebilir.

Hata düzeltme ihtiyacı ancak son zamanlarda önem kazandı, konunun ekonomisi hakkında çok az şey biliniyor. Bu tür kodların kullanılmasında, yukarıda belirtilen azaltılmış etkin kanal kapasitesinin yanı sıra, kodlama ve hataları düzeltmek için ekstra ekipman olacağı açıktır. Bu hususlar nedeniyle, bu kodların uygulamalarının ilk önce yalnızca aşırı koşullar altında gerçekleşmesi beklenebilir. Bazı tipik durumlar şöyle görünür:

A. minimum bekleme ekipmanı ile uzun süreler boyunca gözetimsiz çalıştırma.

B. Tek bir arızanın tüm kurulumu engellediği son derece büyük ve birbiriyle sıkı ilişkili sistemler.

C. Gürültünün sinyal üzerindeki etkisini azaltmanın imkansız veya ekonomik olmadığı durumlarda gürültü varlığında sinyal verme.

Bu durumlar giderek daha sık yaşanıyor. İlk ikisi özellikle büyük ölçekli dijital bilgi işlem makineleri için geçerlidir, üçüncüsü ise diğer yerlerin yanı sıra "sıkışma" durumlarında ortaya çıkar.

İlk olarak uygulanması en muhtemel durumlarda hata tespit ve düzeltme kodlarının tasarlanması ilkeleri bu yazıda verilmiştir. Bu ilkelerin uygulanmasına yönelik devreler, iyi bilinen tekniklerin uygulanmasıyla tasarlanabilir, ancak sorun burada tartışılmamaktadır. Makalenin I. Kısmı, aşağıdaki durumlarda özel minimum artıklık kodlarının nasıl oluşturulacağını gösterir:

A. tek hata tespit kodları

B. tek hata düzeltme kodları

C. tek hata düzeltme artı çift hata tespit kodları.

Bölüm II, bu tür kodların genel teorisini tartışır ve yapılan varsayımlar altında, Bölüm I'deki kodların mümkün olan "en iyi" olduğunu kanıtlar.

**BÖLÜM I**

**ÖZEL KODLAR**

Bu durumlar giderek daha sık yaşanıyor. İlk ikisi özellikle büyük ölçekli dijital bilgi işlem makineleri için geçerlidir, üçüncüsü ise diğer yerlerin yanı sıra "sıkışma" durumlarında ortaya çıkar.

İlk olarak uygulanması en muhtemel durumlarda hata tespit ve düzeltme kodlarının tasarlanması ilkeleri bu yazıda verilmiştir. Bu ilkelerin uygulanmasına yönelik devreler, iyi bilinen tekniklerin uygulanmasıyla tasarlanabilir, ancak sorun burada tartışılmamaktadır. Makalenin I. Kısmı, aşağıdaki durumlarda özel minimum artıklık kodlarının nasıl oluşturulacağını gösterir:

A. tek hata tespit kodları

B. tek hata düzeltme kodları

C. tek hata düzeltme artı çift hata tespit kodları.

Bölüm II, bu tür kodların genel teorisini tartışır ve yapılan varsayımlar altında, Bölüm I'deki kodların mümkün olan "en iyi" olduğunu kanıtlar.

BÖLÜM I

ÖZEL KODLAR

2. TEK HATA TESPİT KODLARI

Aşağıdaki şekilde 1: ikili haneli tek bir hata tespit kodu oluşturabiliriz: İlk n - 1 pozisyona n - 1 basamak bilgi koyarız. n'inci konuma 0 veya 1 yerleştiririz, böylece 1: konumlarının tamamı çift sayıda 1'e sahip olur. İletimdeki herhangi bir tek hata, bir kod sembolünde tek sayıda 1 bırakacağından, bu açıkça tek bir hata tespit kodudur.

Bu kodların fazlalığı, m = n - 1 olduğundan,

Düşük bir artıklık elde etmek için n'nin çok büyük olmasına izin vermeliyiz gibi görünebilir. Ancak, n'yi artırarak, bir sembolde en az bir hata olasılığı artar; ve fark edilmeden geçebilecek çifte hata riski,

da artar. Örneğin, eğer p« 1 herhangi bir hatanın olasılığı ise, o zaman 1/p kadar büyük n için, doğru bir sembolün olasılığı yaklaşık olarak l/e = 0.3679 ... 'dir, çift hata ise l/2e olasılığına sahiptir. = 0.1839...

Sembolün tek bir hataya sahip olup olmadığını belirlemek için yukarıda kullanılan kontrol türü, kağıt boyunca kullanılacak ve parite kontrolü olarak adlandırılacaktır. Yukarıdakiler eşit bir parite kontrolüydü; kontrol pozisyonunun ayarını belirlemek için tek sayıda 1 kullanmış olsaydık, bu tek bir parite kontrolü olurdu. Ayrıca, bir parite kontrolünün her zaman sembolün tüm pozisyonlarını içermesi gerekmez, sadece seçilen pozisyonlar üzerinde bir kontrol olabilir.

3. TEK HATA DÜZELTME KODLARI

Tek bir hata düzeltme kodu oluşturmak için ilk olarak 1t mevcut pozisyonun m'sini bilgi pozisyonları olarak atadık. m'yi sabit olarak kabul edeceğiz, ancak belirli konumlar daha sonraki bir belirlemeye bırakılmıştır. Daha sonra kalan k pozisyonu kontrol pozisyonları olarak atayacağız. Bu k pozisyonundaki değerler, seçilen bilgi pozisyonları üzerinde eşit parite kontrolleri ile kodlama sürecinde belirlenecektir.

Bir an için hatasız veya hatasız bir kod sembolü aldığımızı hayal edelim. k parite kontrollerini sırayla uygulayalım ve parite kontrolünün kontrol pozisyonunda gözlenen değeri atadığı her sefer için 0 yazıyoruz, atanan ve gözlenen değerler uyuşmadığında ise 1 yazıyoruz. bir satırda sola k 0'lar ve l'ler dizisi (parite kontrolleri tarafından atanan değerlerden ayırt edilmek üzere) ikili bir sayı olarak kabul edilebilir ve kontrol numarası olarak adlandırılacaktır. Bu kontrol numarasının herhangi bir tek hatanın pozisyonunu vermesini talep edeceğiz, sıfır değeri sembolde hata olmadığı anlamına gelir. Bu nedenle çek numarası m + k + 1 farklı şeyleri tanımlamalıdır, böylece

ben>m+k+1

k üzerinde bir koşuldur. n = m + k yazarak buluruz Bu eşitsizliği kullanarak, belirli bir n için maksimum m'yi veren Tablo I'i veya aynı şey, belirli bir m için minimum 1t'yi hesaplayabiliriz. çeşitli parite kontrollerinin her biri uygulanmalıdır. Kontrol numarası, parite kontrolleri sırayla uygulanarak ve duruma göre 0 veya 1 yazılarak sağdan sola basamak basamak elde edilir. Kontrol numarası, bir kod sembolündeki herhangi bir hatanın konumunu vereceği için, ikili gösteriminin sağında 1 olan herhangi bir pozisyon, ilk kontrolün başarısız olmasına neden olmalıdır. Bulduğumuz çeşitli tam sayıların ikili biçimini incelemek

1 = 1

3 = 11

5 = 101

7 = 111

9 = 1001

Vb.

aşırı sağda 1 var. Bu nedenle ilk eşlik kontrolü 1, 3, 5, 7, 9, .... pozisyonlarını kullanmalıdır.

Tam olarak benzer bir şekilde, ikinci eşlik kontrolünün, ikili gösterimlerinin sağından ikinci basamak için 1'leri olan konumları kullanması gerektiğini bulduk.

2 = 10

3 = 11

6 = 110

7 = 111

10 = 1010

11 = 1011

Vb.,

üçüncü parite kontrolü

4 = 100

5 = 101

6 = 110

7 = 111

12 = 1100

13 = 1101

14 = 1110

15 = 1111

20 = 10100

Vb.

H, her bir parite kontrolü için hangi pozisyonların bilgi içereceğine ve hangi kontrolün olacağına karar verir. Aşağıdaki tabloda verilen kontrol konumları için 1, 2, 4, 8, ... konumlarının seçimi, kontrol konumlarının ayarını birbirinden bağımsız yapma avantajına sahiptir. Diğer tüm pozisyonlar bilgi pozisyonlarıdır. Böylece Tablo II'yi elde ederiz.

Yukarıdaki teorinin bir örneği olarak, onu yedi konumlu bir kod durumuna uyguluyoruz. Tablo I'den n = 7, m = 4 ve k = 3 için buluyoruz.

Tablo II, birinci parite kontrolünün 1, 3, 5, 7 konumlarını içerdiğini ve birinci konumdaki değeri belirlemek için kullanıldığını görüyoruz; ikinci parite kontrolü, 2, 3, 6, 7 konumları ve ikinci konumdaki değeri belirler; ve üçüncü parite kontrolü, 4,5,6, 7 konumları ve dördüncü konumdaki değeri belirler. Bu, 3, 5, 6, 7 konumlarını bilgi konumları olarak bırakır. 3, 5, 6, 7 konumlarını kullanarak olası tüm ikili sayıları yazmanın ve ardından 1, 2, 4 kontrol konumlarındaki değerleri hesaplamanın sonuçları Tablo III'te gösterilmektedir.

Böylece yedi konumlu tek hata düzeltme kodu, 16 kod sembolünü kabul eder. Elbette 2^7 - 16 = 112 anlamsız sembol vardır. Bazı uygulamalarda, bir kod sembolü veya bir kod sembolü artı tek bir hata olarak tamamen sıfır kombinasyonundan kaçınmak için ilk sembolün koddan çıkarılması istenebilir, çünkü bu mesajsız olarak karıştırılabilir. Bu hala 15 faydalı kod sembolü bırakacaktır.